LaTeX 入門

平成 29 年 10 月 3 日一部改定

平野 照比古

この文書についてお問い合わせは下記のメールまでご連絡ください。

©2006/9-2017/10 平野照比古

e-mail:hilano@ic.kanagawa-it.ac.jp
URL:http://www.hilano.org/hilano-lab

目 次

第 1 章　TEX とはなにか　1
1.1　開発の由来 1
1.2　TEX のシステムについて 1
1.3　他の TEX のシステム 3
1.4　LATEX に関する素朴な質問 4

第 2 章　LATEX のインストールと実行　7
2.1　LATEX システムが動くまで 7
2.1.1　LATEX のインストール 7
2.1.2　GhostScript と GhostView のインストール 11
2.1.3　`dviout` のインストールと設定 14
2.2　LATEX を使用する 18
2.2.1　コマンドプロンプトのショートカットの作成 18
2.2.2　ファイルの準備 19
2.2.3　`dvi` ファイルの作成 20
2.2.4　組版結果の表示 21
2.2.5　エラーへの対処 22

第 3 章　LATEX ファイルを作る　25
3.1　LATEX ファイルの構造 25
3.2　LATEX の約束事 25
3.2.1　特別な扱いの文字 26
3.2.2　空白と段落 26
3.3　フォントを変える 27
3.4　原稿を見る 28
3.5　いろいろな文書の形式 30
3.5.1　箇条書きの書き方 30
3.5.2　中央ぞろえ 30
3.5.3　右ぞろえ 31
3.5.4　脚注 31
3.5.5　ページ内にページを作る 31
3.5.6　プログラムリストを表示する 32
3.5.7　強制改ページ 34

3.5.8 番号が自動でつく環境を定義する 34
3.6 その他知っておくと便利なこと 35
3.6.1 番号付け 35
3.6.2 入力ファイルの分割 36
3.6.3 まとまった部分をコメントにする 37
3.6.4 印刷範囲を大きくする 37
3.6.5 均等割付 37
3.6.6 アクセントのついた文字 39
3.6.7 特殊な文字 39

第 4 章 表組み 41
4.1 表組みの基本 41
4.2 array パッケージ 44
4.3 表組みの Tips 45
4.3.1 部分的に横線を入れる 45
4.3.2 行方向にいくつかの項をまとめる 46
4.3.3 列方向にいくつかまとめる 47
4.3.4 表の罫線の幅を変える 47
4.4 table 環境 48
4.5 ページをまたぐ表を作成する 49

第 5 章 図の取り扱い 53
5.1 picture 環境 53
5.1.1 picture 環境の基礎 53
5.1.2 繰り返しの出力 55
5.2 graphicx パッケージ 56
5.3 figure 環境 60
5.4 PostScript ファイルへの変換 60
5.4.1 jpeg ファイルを PostScript ファイルに変換する 60
5.4.2 コンピュータの画面のキャプチャー 61

第 6 章 数式の書き方 63
6.1 数式モードの違い 63
6.2 簡単な数式の書き方 63
6.2.1 上付きと添え字 64
6.2.2 分数の書き方 64
6.2.3 総和などの書き方 65
6.3 別立ての数式の書き方 65
6.4 いろいろな記号の書き方 66
6.5 内容に応じたおおきな括弧を作る 66

第 7 章　マクロの書き方 **69**
7.1 TEX のマクロとは 69
7.2 マクロの書き方 69
7.3 マクロにおける空白について 70
7.4 マクロの書き方–発展 70
7.5 \newif 71
7.6 比較のマクロ 71
7.7 \ifcase 74

第 8 章　相互参照と索引の作成方法 **75**
8.1 相互参照 75
8.1.1 相互参照の基礎 75
8.1.2 文献の参照 75
8.1.3 相互参照で表示される値について 76
8.1.4 Bad Section Format(文書とは無関係です) 76
8.1.5 Good Section Format(文書とは無関係です) 77
8.2 索引の作成 78
8.2.1 索引作成の基礎 78
8.2.2 索引作成の詳細 78
8.3 hyperref パッケージ 80
8.4 この文書での索引の作成方法 82

第 9 章　LATEX でプレゼンテーションをする **83**
9.1 LATEX とプレゼンテーション 83
9.2 beamer の特徴 83
9.3 原稿の構成 84
9.3.1 プリアンブル 84
9.3.2 本文 84
9.4 プレゼンテーションファイルの作成法 85
9.4.1 ハンドアウトの作成 86
9.5 ページの体裁の変更 87
9.6 オーバーレイの指定 87
9.6.1 箇条書きの項目を順番に表示する 87

付 録 A　PS ファイル **91**

付 録 B　LATEX の参考書について **93**
B.1 LATEX の入門書 93
B.2 TEX のマクロの書き方の参考書 93
B.3 PostScript の参考書 94

索引 **97**

第1章　TEX とはなにか

1.1　開発の由来

TEX は [5] の著者 D. E. Knuth がその著作の印刷結果に満足しなかったために Knuth 自身が開発した組版システムです。組版システムとは出版物にするための体裁を整えることを目的とするものです。Knuth 自身 [4, Preface] において次のように述べています。

> This is a handbook about TEX, a new typsettiang system intended for the creation of beautiful books—and especially for books that contain a lot of mathematics.

Knuth 自身はこの開発に 10 年ほどの年月を要し、システムが安定したのでこれ以上の開発はしないと宣言して、長年の懸案であった The Art of Computer Programming の著書の完成を目指すといっています ([5, Preface,xi])。なお、TEX の発音については Knuth 自身が解説しています ([4, p.1]) のでそれを参考してください。

1.2　TEX のシステムについて

TEX はコンピュータ上に組版に特化した環境を作成します。最近の用語で言えば TEX VM(Virtual Machine) を構成するということになります。この TEX VM は基本的な組版の処理しかできません。このままの環境を直接利用するのはユーザーに負担がかかります。そこでより使いやすい命令を備えたパッケージが開発され、ふつうそのパッケージを利用して原稿を書きます。この関係はちょうどプログラムを書くときにコンピュータが直接理解できる機械語 (またはアッセンブラー) で書く代わりに別の言語でプログラムを書いて計算をさせるという関係とよく似ています。

Knuth が用意したパッケージは plain TEX とよばれています。Knuth の著作 [5] はこの解説書です。TEX は次のような特徴を持っています。

- TEX はいわゆるタグ (に相当するマクロ) を用いて文書を記述します。したがって、記述している段階では文書がどのように整形されるかわかりません。文書はいわゆるテキストエディタで作成します。ファイルの拡張子は `.tex` とするのが普通です。

- 作成した文書を TEX のシステムを通すことで実際の組版の結果が得られます。組版の結果、作成されたファイルの拡張子は `.dvi` となります。これは Device Independent の略です。つまり、表示機器によらない形式のファイルが出来上がります。このことを TEX ファイルをコンパイルするという場合があります。

- TEX は文字の字形は無視してそれを囲む箱として捕らえ、それらを並べて単語を作成し、それらの間隔をアルゴリズムにしたがって配置します。配置の基準は段落やページ単位となっているのが特徴です。

- このファイルを実際に表示するのは TEX のシステムにはありません。ディスプレイに表示するためのプログラムは一般にプレビューアと呼ばれます。Windows では dvioutが標準です。最近ではプレビューアから印刷できるようになっているのが普通です。TEX にはプレビューアに対する命令を直接指示できる機能があります。この機能により TEX に備わっていない機能を追加することが可能となっています。

- TEX は数式を書くモードと普通の文章を書くための二つのモードがあります。したがって、数式は普通の文章とは異なるアルゴリズムで組版をするので非常にきれいに書くことができます。数式の構造を与えるだけで自動で適切な配置をします [1]。

その後、L. Lamport が plain TEX をもとに章や節の番号の自動付け、目次や索引の作成機能を付け、より構造化された文書を plain TEX より容易に作成できるシステム LATEX を作成しました [6]。LATEX は plain TEX を基にして各種の命令 (マクロ) を追加したものです。また、組版の規則をより追加しています。LATEX の登場により TEX の普及が加速したことは間違いありません。

LATEX は次のような特徴を持っています [2]。

- 文書クラス [3] の導入
  文書はその使用目的において独自の組版の規則があります。たとえば、出版される書籍では次のような特徴があります。

  - 新しい章は奇数ページから始まる。
  - 右側と左側のページではページの上部にくるタイトル (ヘッダー) が異なる。
  - 偶数ページと奇数ページで左右の余白が異なる。

  これに対し、レポートなどでは普通片面に印刷されるので偶数ページと奇数ページの体裁を区別する必要は普通ありません。組版の各種パラメータを設定するのが文書クラスの役目です。

  文書クラスを作成するのは編集者の仕事です。著者は内容の記述に専念できます。

- 章などの番号の自動設定と目次の作成の自動化
  章や節の番号は自動的に増えながらシステムが順番に付けます。これを元に目次の体裁なども自動で整えられます。途中で章や節を増やしたり減らしたり順序を交換しても番号は自動的に修正されます。

- 相互参照の自動化
  文書の記述で前のほうや後のほうを参照するときにラベルを付けておくことで自動的に番号

[1] TEX の数式表現は形式的なものです。分数の形をしたものはすべて同じ形の入力をすれば十分です。入力した数式の意味を理解して計算をしてはくれません。

[2] 筆者はこの初版の中で文書は構造化されているべきであるという記述に感銘を受けた記憶があります。ただ、XML の観点から見ると構造化はの概念はすばらしかったのですが、すべての記述が構造化されていなかった残念です。

[3] LATEX ではスタイルファイルと呼ばれましたが LATEX 2ε では文書クラスに変更されました

の参照が行われます。また、図や表の番号も自動的に付けられ、その番号や出てきたページを参照できます。
これらの番号の表示形式は自由に設定することができます。

- 各種環境による文書の構造化が可能
  箇条書きや番号のついた箇条書きが簡単に記述できます。また、数式に関する記述を簡単にする環境があります。もちろん、一般的な環境を定義するための環境や、通し番号を付けられる環境を新たに定義することができます

- 表の作成が楽
  環境のひとつとして表を作成する環境があります。表の作成は罫線を引いてするのではなく、表の構造を指定します。LATEX はその構造に基づき項目の間隔などを自動で調整します。行列なども表と同じ形式で表すことができます。

- 数式の記述が楽
  TEX の開発動機にもあったように元来の TEX でも数式はきれいにかけますが LATEX では各種パッケージも開発されていろいろな数式やそれに現れる記号などが準備されています。
  数式に対しても参照するための番号を自動で付けます。

- マクロの定義の簡略化
  TEX でも記述を簡単にするための方法としてマクロと呼ばれる新しい関数を定義する方法が用意されています。LATEX ではこのマクロの定義方法に柔軟性を持った記述ができるようになっています。

- 索引の作成が容易
  索引を作る機能があります。LATEX から出されたデータを下に並べ替えや索引の体裁を整えるためのツールがついています。

現在の LATEX は LATEX 2ε と呼ばれています。LATEX3 へのつなぎのバージョンなのですが 10 年以上たったのにまだ正式バージョンが出ていません。

1.3 他の TEX のシステム

LATEX にはいろいろなバージョンがあります。元来の TEX は日本語対応になっていませんでした。日本語対応にした TEX(LATEX) のひとつとしてアスキー社が開発した pTEX があります。このテキストも pTEX(正確には pLATEX) で書かれています。LATEX にはいろいろなパッケージが開発されていていろいろなものの組版が可能になっています。化学式を書いたり、楽譜を書いたりするパッケージがあります。詳しくは [8] や [9] などを見てください。

また、Unicode 対応のものや直接 pdf ファイルを出力できる pdfTEX も開発されています (残念ながら日本語は扱えません)。

なお、TEX の商標はアメリカ数学会 (AMS) が所有しています。AMS は数学の論文を記述するために $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX を作成しています。これには数学で使われるいろいろな記号が追加されています。

これらの成果を LATEX に取り入れたのが $\mathcal{A}\mathcal{M}\mathcal{S}$-LATEX です。パッケージを追加することでこれらのものを利用できます。

1.4　LATEX に関する素朴な質問

LATEX は有償ですか

いいえ。LATEX などの商標はアメリカ数学会が所有していますが原則として無償のソフトウェアです。LATEX で利用する標準的なソフトウェアも無料です。

最新版はどこで手に入りますか

各プラットフォームで最新版を配布するサイトがあるのでそちらを利用してください。Windows 版は近畿大学の角藤さんが配布しているものがよいでしょう。詳しくは第 2 章を参照してください。

LATEX はどのようにして学べばよいですか

入門書を 1 冊購入してある程度なれることが必要です。なかなかよい本がないのが現状です。[7] がかなりの部分をカバーしています。書籍の紹介などは巻末の付録 B を参考にしてください。

LATEX の文書を作成するツールは何ですか

Windows で標準についてくる「メモ帳」のようなテキストエディタと呼ばれる単純なもので作成できます。詳しくは次の章を参考にしてください。

Java の統合環境として有名な Eclipse でも TEX に対応した環境が作れるもの [4] があります。

このテキストではコマンドプロンプトから pLATEX を実行することを前提にすべて無料で環境が整えられることを目標に書かれています。

エラーの意味が良くわからないのですが

いくつか代表的なものを 2.2.5 に載せましたので参考にしてください。

インストールされていないパッケージはどこにありますか

`CTAN`(ComprehensiveTEX Archive Network) の中を探してください。`CTAN` はいろいろなところにミラーされています [5] ので近いところを探してください。たとえば、日本では ring サーバー [6] がミラーしています。目的のパッケージの所在はそのにある `FILESbyname` を中から所在の場所を探してください。このファイルは単純なテキストファイルです。

新しく入れたパッケージが見つからないというメッセージがでます

配布されたパッケージをいれたあとファイルのデータベースの更新をする必要があります。

```
mktexlsr
```

を実行してください。

[4] `http://eclipsewiki.net/eclipse/?TeX`

[5] 作成者が配布しているサイトの内容を定期的にコピーして公開することを元のサイトをミラーするといい、そのようなサイトをミラーサイトといいます。

[6] `ftp://ftp.ring.gr.jp/pub/text/CTAN/`

LaTeX を利用してプレゼンテーションができますか

可能です。次のような方法を単独、または組み合わせてする方法があります。

- LaTeX にある `slides` というプレゼンテーションに特化した文書クラスを利用する。
- Windows のプレビューアである `dviout` にあるプレゼンテーション用のモードを利用する。
- 標準のスタイルファイルより大きなフォントが利用できる `foils` パッケージを用いる。筆者は [10] の解説を参考にして LaTeX ファイルを作成し、最終的に pdf ファイルに直してプレゼンテーションをしていました。
- `beamer` クラスを用いる。比較的新しい文書クラスです。2008 年度からはこれを用いてプレゼンテーションをすることにしました。第 9 章で解説します。

LaTeX を利用して HTML 文書が作成できますか

`latex2html` というシステムがあります。日本語用 Windows 用のシステム [7] を公開しているところがありますので参考にしてください。このページそのものが `latex2html` で変換されたものです。

- 節単位で異なるページに変換されます。ページ間のリンクは自動で作られます。
- 数式などは画像に変換されて表示されます。そのために、インストールが少し面倒になっています。
- 手の込んだマクロを利用していると変換ができない場合があります。

同一の文書から LaTeX 用の文書や HTML 文書を作成する SmartDoc[8] という XML の規格もあります。Java の実行環境 (SDK) が必要です。

- Java で書かれた各種のフォーマットへの変換のプログラムが提供されています。LaTeX に変換された文書はこの後で LaTeX にかけて変換します。
- XML の要素名は LaTeX や HTML からの借用されたものが多いです。したがって、LaTeX と HTML のタグを知っていると便利です。
- 厳密な XML の規格に従っていますので要素には必ず終了タグが必要ですし、対応するタグは大文字小文字含めて同じでなければなりません。

まお、筆者の大学のホームページは SmartDoc で書かれています。

また、[10] も参考になります。

LaTeX で作成した文書を LaTeX をインストールしていない人に配布したいのですが...

`dvi` ファイルだけで渡すと問題が生ずる (フォントがない、図のファイルをつけるのを忘れる) ので `pdf` に直して渡すことをお勧めします。第 2 章を見てください。

[7] `http://takeno.iee.niit.ac.jp/ shige/TeX/latex2html/l2h-win.html`
[8] `http://www.asamioffice.com/ja/2010/smartdoc/download/index.html`

pdf ファイルを作成するのに Adobe Acrobat Pro DC はいらないのですか

不要です。TEX の配布システムの中に TEX の文書を組版した後にできるファイルから pdf に変換するツールがあります。詳しくは 2.2.4 を見てください。

第2章　LaTeXのインストールと実行

LaTeX をインストールするためにはたとえば [14] の付録にある CD からインストールするという方法もありますが、バージョンアップやインストールされていないパッケージを追加する方法がわからなくなるという欠点もありますのでここではネットからダウンロードしてする方法について解説します。

なお、このテキストで \ と表示されている文字はすべて ¥ で入力してください。一部のエディタでは¥ を入力してもで \ と表示さるものもあります。

2.1　LaTeX システムが動くまで

2.1.1　LaTeX のインストール

ここでの解説は自力で TeX のシステムを Windows 上にインストールする方法を解説します。福岡教育大学の藤本研究室からはより簡単なインストーラー付で TeX の環境を構築するパッケージが配布されています。`http://ww1.fukuoka-edu.ac.jp/~fujimoto/tex.html` を参照してください。ここで配布されるシステムはこの節で紹介する方法と次の点で異なっています。

- 作成するファイルの文字コードは UTF-8 です。
- LaTeX のキーワードを認識して色が変わるエディタが付属しています。
- 組版の結果は pdf ファイルになります。
- pdf ファイルを閉じなくでも最新の結果が表示されます [1]。

LaTeX システムのダウンロード　Windows における LaTeX のパッケージとしては関西大学の角藤さんが配布しているパッケージが標準です。ます、配布されている場所から必要なものをダウンロードします。配布サイトとしては ring サーバー [2] を利用します。それぞれのサイトではこれらのファイルをミラーしているかもしれません。できるだけ近いところから入手しましょう。比較的メンテナンスがされているので新しいファイルだけダウンロードできるような `ftp` クライアントを用意するのがよいでしょう。すべてのファイルが必要ではないのですがここでは上記の場所にあるファイルをすべてダウンロードしましょう。ダウンロードする場所は新しくフォルダを作成しておくことをお勧めします。ここでは `C:\download\pTeX` としておきます。

[1] このシステムから印刷をする場合には Adobe Reader で pdf ファイルを開きます。このとき、ファイルを開いたままにしておくと、修正した後、pdf ファイルが書き込めないというエラーが出ます。これは Adobe Reader が実行されている間、対象のファイルを開きっぱなしにしているためです。

[2] `ftp://ftp.ring.gr.jp/pub/text/TeX/ptex-win32/current/`

インストールの準備 次に、コマンドプロンプトから次のコマンドを順に打ち込みます。

```
cd \
mkdir usr
cd usr
mkdir local
cd local
```

これによりカレントディレクトリ (プログラムを実行する基準の場所) が C:\usr\local になります。

次に、先ほどダウンロードしたファイルのうち texinstXXXX.zip のファイル (XXXX の部分は数字でその年の西暦年になっています) をエクスプローラーを利用して C:\usr\local にコピーし、解凍します (右ボタンでここに解凍を実行)。

インストールの実行 解凍されたフォルダー内に texinstXXXX.exe があることを確認します。エクスプローラーからでもよいのですが、先ほどのコマンドプロンプトの上で

```
dir
```

と打ち込んで上記のファイルがあるか確認しましょう。あることが確認できたらいよいよインストールの開始です。

```
texinstXXXX C:\download\pTeX
```

と打ち込みます。 XXXX の部分は数字に直してください。画面に表示が流れますので終了するまで一休みしてください。

環境変数の設定 TeX のシステムがコマンドプロンプトから実行できるように環境変数 [3]PATH に C:\usr\local を追加します。作業は次のようにします。

- スタート ⟹ 設定 ⟹ コントロールパネル で「コントロールパネル」を開きます。
- コントロールパネルの表示方法がカテゴリー表示になっている場合 (図 2.1) には「パフォーマンスとメンテナンス」をクリックしてそこからシステムをクリックします。カテゴリー表示でない場合 (クラシック表示) の場合はそこで「システム」をクリックします。
- 図 2.2 で「詳細設定」のタブをクリックします。
- 図 2.3 において最下部中央にある「環境変数」のボタンをクリックします。
- 図 2.4 において上部にあるユーザーの環境変数の「新規」のボタンをクリックします (下のほうで設定してもかまわないのですが問題が発生したときに元に戻せないと困るのでここではユーザー変数のほうに設定することにします)。
- 図 2.5 が開きますので「変数名」に PATH を値には C:\usr\local\bin を設定します (図 2.6)。

[3]環境変数とはあるプログラムを実行するときに特別の指定がない場合の値 (デフォルト値) を指定するときなどに利用されます。環境変数 PATH は指定されたプログラムが現在のディレクトリにない場合に探してくるためのフォルダーのリストを与えます。このことをパスを通すといったりします。

図 2.1: コントロールパネルの表示 (カテゴリー表示)

図 2.2: システムのプロパティ

- 「OK] ボタンを押して設定を終了します。

この作業が終了したらコマンドプロンプトを終了し、再びコマンドプロンプトを実行して上で行った環境変数の追加を反映させます。。

```
platex
```

と打ち込んで

```
This is e-pTeX, Version 3.14159265-p3.7-160201-2.6 (sjis) (TeX Live 2016/W32TeX)
```

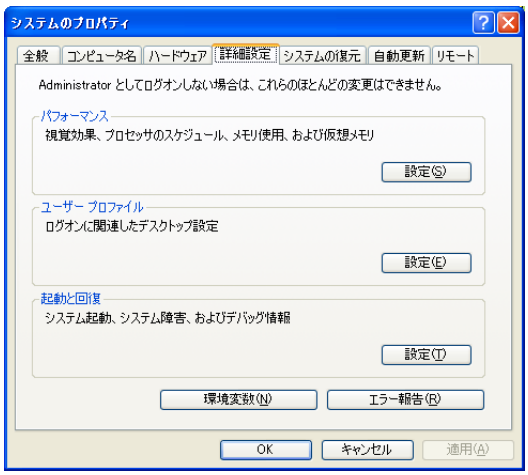


図 2.3: システムのプロパティ

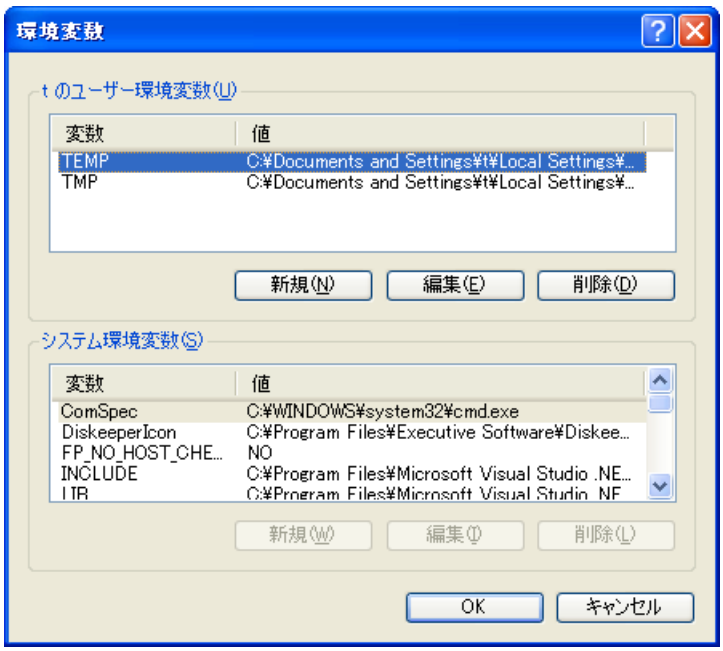


図 2.4: 環境変数の設定

```
(preloaded format=platex)
 restricted \write18 enabled.
**
```

のような表示が現れればインストールが正しく行われ、環境変数が正しく設定されています。ここでは入力ファイルを指定していないので処理が途中で止まります。`Ctrl+C` でプログラムの実行をとめてください。

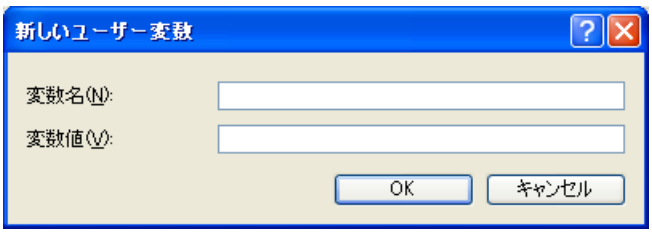


図 2.5: 新規の環境変数の設定

なお、インストール後には bin, contrib, share などのフォルダーが作成されます。

2.1.2 GhostScript と GhostView のインストール

LaTeX で画像を取り扱う方法はいろいろありますが、最終的に作成した文書は pdf ファイルにすることを考えると画像のファイルは eps ファイルにしておくのが賢明と思われます。

Ghostscript は Adobe 社が開発したプログラミング言語 PostsCript のインタープリターです。ライセンス形態により AFPL ライセンスと GPL ライセンスの二つがあります。

PostScript とは

PostScript[4] とは Adobe 社が開発したプログラミング言語で、言語の仕様にグラフィックスの環境を含んでいるのが特徴です。一般にはページ記述言語 (1 ページの中にオブジェクトをどの位置に配置するかという情報を記述することができる) と述べられています。標準の拡張子は.ps です。また、1 ページだけからなる PostScript のデータに含まれる内容の配置位置 (オブジェクトを包み込む箱–バウンディングボックス) が記述されている Postcript ファイルを Encasplated PostScript と呼びます。その拡張子は .eps です。プログラムの書き方は文献 [1](その本の表紙の色から Blue Book とよばれています) を、言語の完全な仕様を見たければ文献 [2](その本の表紙の色から Red Book とよばれています) を参照してください。この文献は Adobe 社のホームページ [5] からも得られます。

なお、Postscript からプログラミングの部分を省略したのが pdf ファイルです。

Adobe 社の Illustrator で作成したファイルは ps 形式で保存できます。

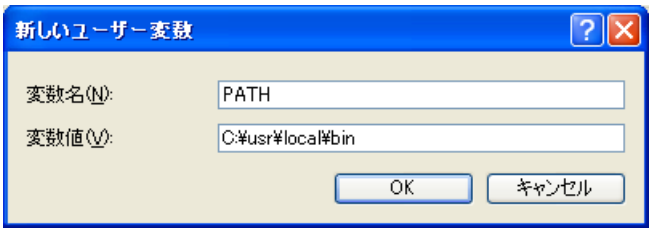


図 2.6: 環境変数の設定

[4] http://www.adobe.com/products/postscript/
[5] http://www.adobe.com/products/postscript/pdfs/PLRM.pdf

GhostScript のインストール

GhostScript のインターネットサイトでは日本語が使えない (表示できない) パッケージが正式なものです。日本語の表示ができるものはたとえば ring サーバー[6] にあります。`gsXXXwYYfull-gpl.exe`(XXX は数字で 2016 年 4 月 4 日では 919 になっていました。また YY の部分は 32 または 64 で 32 ビット OS と 64 ビット OS に応じて変えてください) をダウンロードしてこのファイルを実行します。実行すると図 2.7 が表示されます。「Install」ボタンを押すとインストールが開始されます。イン

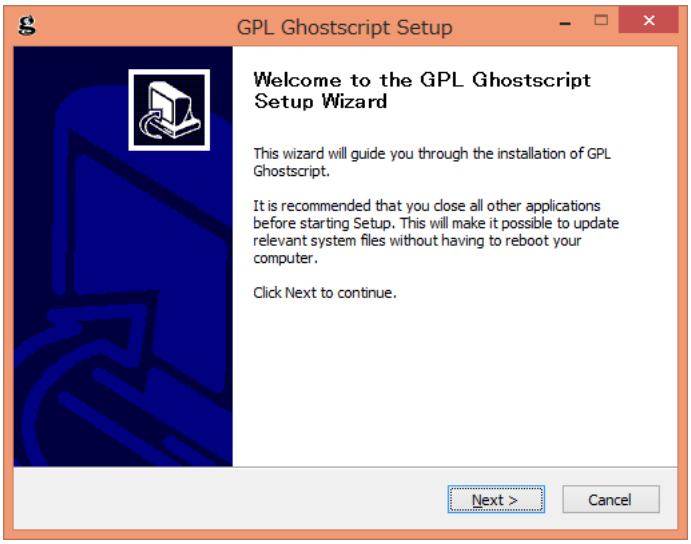


図 2.7: ghostscript のインストール開始画面

ストールの場所を設定する画面 (図 2.8) が表示されますが特別な事情がない限りデフォルトの場所でかまいません。

このあとインストールした GhostScript の実行ファイルが存在するディレクトリ (`C:\gsXXX\gs\bin`) を環境変数 `PATH`(XXX の部分はバージョン番号) に追加します。この部分を記述する前に `;`(セミコロン) で区切ってください。コマンドプロンプトを立ち上げなおして

```
gswin32
```

または

```
gswin32
```

と打ち込んで GhostScript が起動することを確認します。図 2.9 のようなウインドウが開いけばインストールは成功です。[7]

[6] `ftp://ftp.ring.gr.jp/pub/text/TeX/ptex-win32/gs`
[7] 実はこのほかに図を表示するためのウィンドウも表示されています。

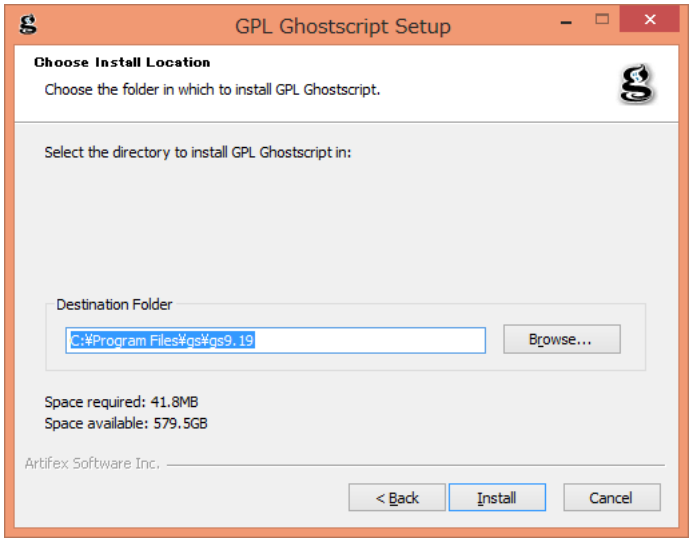


図 2.8: ghostscript のインストール場所の設定

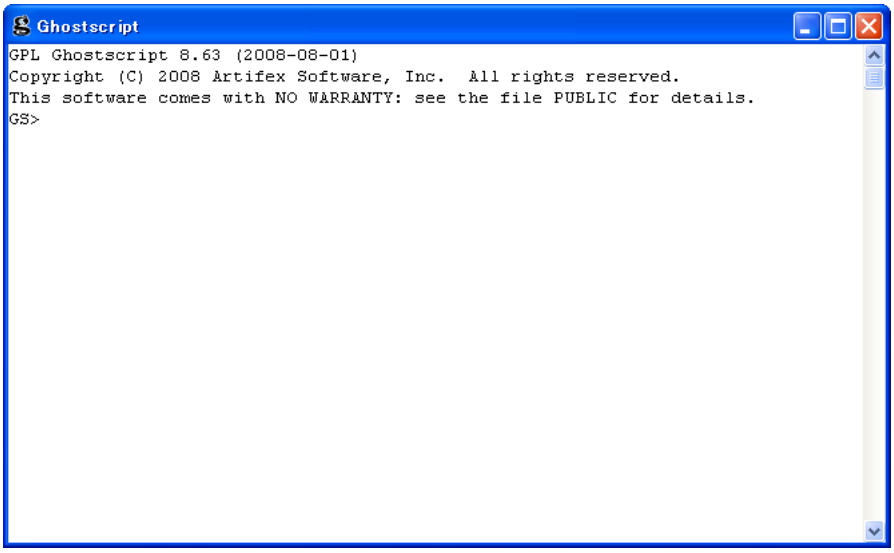


図 2.9: gswin32 の起動時の画面画面

GhostView のインストール

TeX で GhostScript を利用する場合に、GhostView は必要ありませんが、自分で PostScript を用いて図を書いたときのプレビューアとして利用することも可能ですし、作成した PS ファイルを eps ファイルに変換する機能がありますので、ここに説明しておきます。

1. ダウンロードしたファイルをダブルクリックする。
2. あとは指示されたとおりにボタンを押していきます。メッセージは日本語になっていません

ので「English」を選択してください。インストールするフォルダーもデフォルトでかまいません。

インストールが終了したら「gsview」のアイコンをクリックして起動し、ghostscript(gswiew ではない) のフォルダにある example や kanji フォルダ内のファイルを表示できることを確認してください。

2.1.3 dviout のインストールと設定

先ほど LaTeX のシステムをダウンロードしたところから dviout のシステムをダウンロードします。場所はここ [8] です。texXXXw.zip(XXX は数字です。2015 年 10 月 5 日現在 318 になっています) を展開したのち、出来上がったフォルダをインストールするフォルダーに移動します。コマンドプロンプトから dviout を実行したいのであれば環境変数 PATH を変更しなくてすむように C:\usr\local\bin を指定するとよいでしょう (図 2.10)。

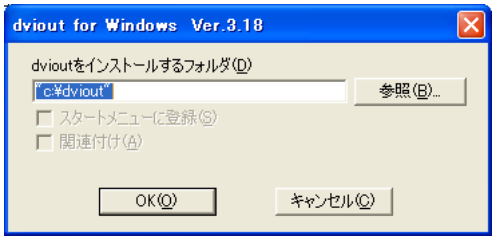


図 2.10: dviout のインストール開始画面

「OK」のボタンを押すと dviout が起動します (図 2.11)。

メニューから「option」$\Longrightarrow$「setup parameters」を選ぶと図 2.12 が開きます。なお、はじめて dviout を起動するとパラメータの設定をするかどうかのウィンドウが開きます。

[8] ftp://ftp.ring.gr.jp/pub/text/TeX/dviout

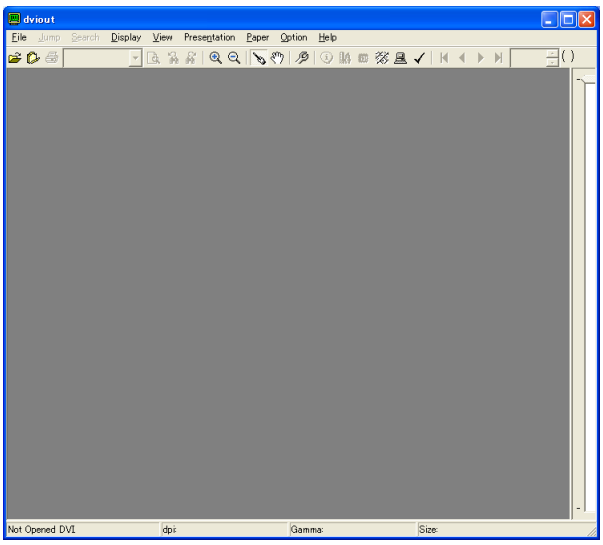


図 2.11: dviout の開始画面

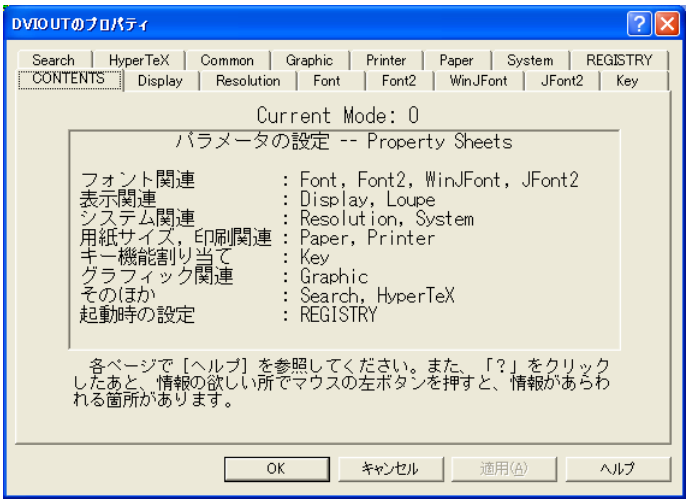


図 2.12: dviout の設定 (1)

「Font」タブをクリックしすると図 2.13 が開きます。

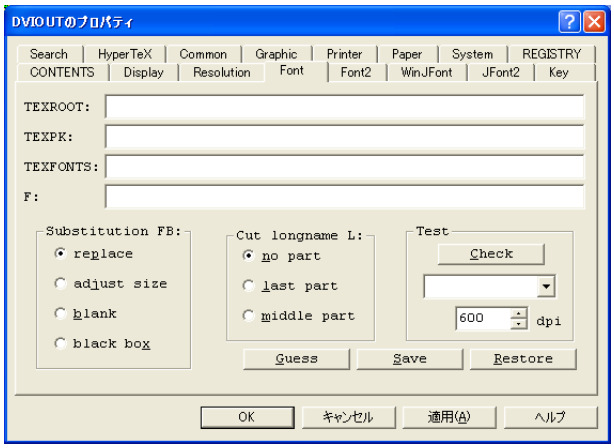


図 2.13: dviout の設定—フォントパスの設定

ここで「Guess」ボタンを押すと図 2.14 のメッセージボックスが開きます。「ok」を押すと適切な値が設定されます。

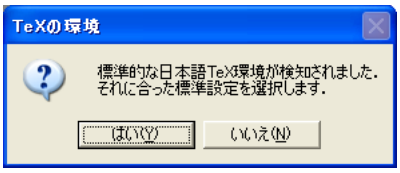


図 2.14: dviout の設定—日本語 TeX の選択

図 2.13 の「Save」ボタンを押してここでの設定は終了です。

次に、「Font2」タブをクリックすると図 2.15 が開きます。

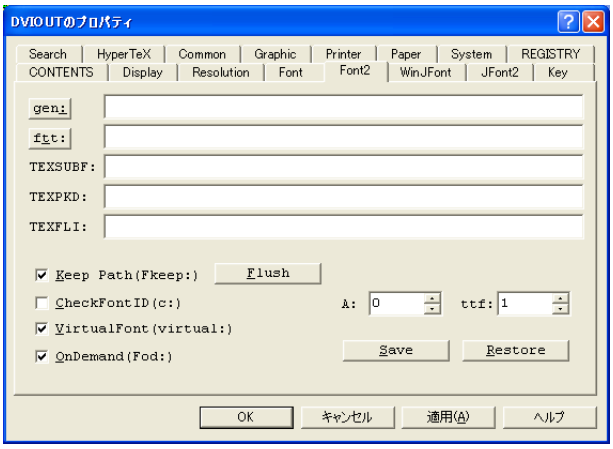


図 2.15: dviout の設定—フォントの生成の設定

「gen」ボタンを押すと図 2.16 のメッセージボックスが開きます。

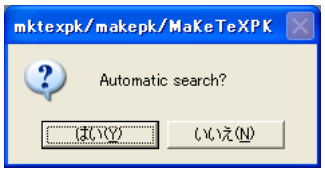


図 2.16: dviout の設定—automatic search の選択

「ok」を押すと適切な値が設定されます。
図 2.15 で「ftt」ボタンを押しますと図 2.17 のメッセージボックスが開きます。

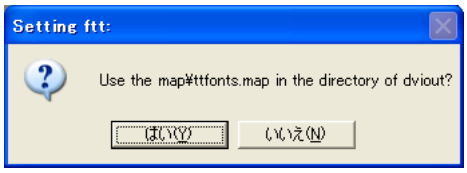


図 2.17: dviout の設定—トゥルータイプフォントの定義の選択

「ok」を押すと適切な値が設定されます。
「Save」ボタンを押してここでの設定は終了です。
次に、GhostScript をインストールしている場合には「Graphic」のボタンを押します (図 2.18)。
ここで左中央部にある「gsx」のボタンを押すと 図 2.19 のメッセージボックスが開きます。「はい」のボタンを押しますと GhostScript の呼び出す値が設定されます。

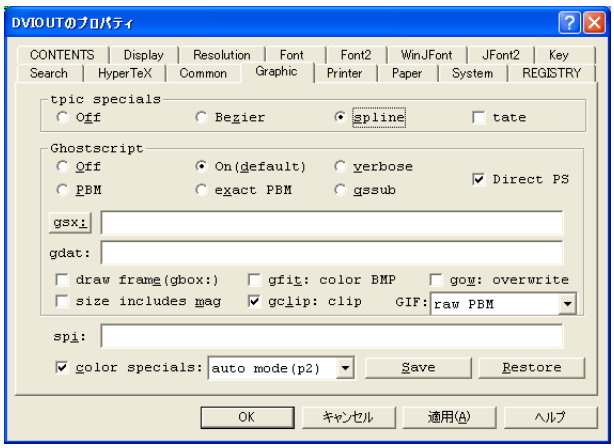


図 2.18: dviout の設定—GhostScript の選択

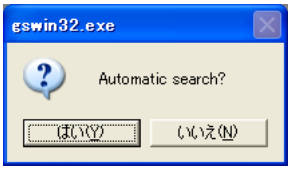


図 2.19: dviout の設定—自動設定の選択

2.2 LaTeX を使用する

2.2.1 コマンドプロンプトのショートカットの作成

コマンドプロンプトを開くと一般には \Document and Setting の下にあるログインしているユーザーのフォルダーがカレントディレクトリになります。以後の作業でこの場所にファイルを作成するのであればこのままでもよいのですが別の場所で作業を行いたい場合に備えてコマンドプロンプトのショートカットを作成します。

1. スタート $\Longrightarrow$ プログラム $\Longrightarrow$ アクセサリを開き、コマンドプロンプトの上で右クリックし、「コピー」を選択します。
2. デスクトップの上で右クリックして「貼り付け」を選択します。
3. 新しくできたショートカットの上で右クリックし「プロパティ」を選択します。(図 2.20)
4. ショートカットのタブを選択し、中ほどにある作業ホルダのなかに、作業するフォルダを記入します。

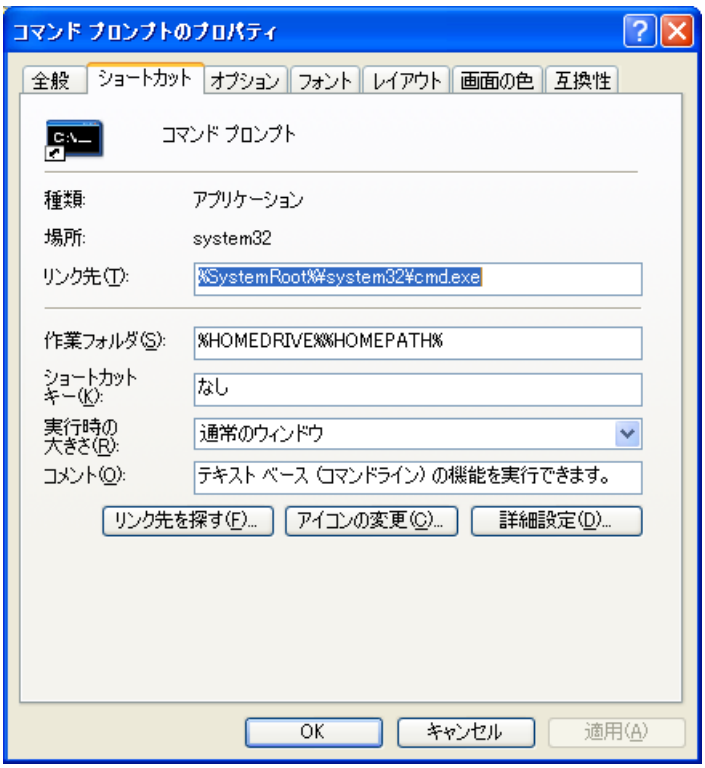


図 2.20: コマンドプロンプトの設定

2.2.2 ファイルの準備

LaTeX のファイルはテキストファイルですので通常のテキストエディタ (Windows のメモ帳) で作成できます。また、LaTeX のソースコードを入力するのに便利な機能を持つものもあります [9]。メモ帳などで次のようなファイルを打ち込んで保存してください。この本文で \ となっている部分はメモ帳などでは ¥ で表示されます。

```
\documentclass{jarticle}
\begin{document}
はじめての \TeX ファイルです。
\end{document}
```

ファイルの拡拡張子は `.tex` としてください。なお、メモ帳の場合は保存するときにファイルの種類を「すべてのファイル」に変更してからファイルを保存してください。ここでは保存したファイル名を `foo.tex` とします。

なお、出力結果を pdf ファイルにしたとき、目次の部分はしおりに反映されます。通常の形式では文字コードがあっていないために文字化けを起こします。それを避けるために、次のコードを

[9] 筆者は Unix 系でのテキストエディタの 2 大勢力のうちのひとつである Emacs を Windows に移植した NTEmacs[10] に LaTeX におけるギリシャ文字や各種の記号が簡単に入力でき、コマンドや環境の記述が簡単になる YeTeX(Yet Another TeX Mode)[11] をインストールして使用しています。

\documentclass{jarticle}に挿入してください。

```
\ifnum 42146=\euc"A4A2
\AtBeginDvi{\special{pdf:tounicode EUC-UCS2}}\else
\AtBeginDvi{\special{pdf:tounicode 90ms-RKSJ-UCS2}}\fi
```

新規に文書を作成するたびにこのコードを入力するのが面倒であれば、この部分だけ入力したファイル (ファイル名を header.tex とします) を作成し、次のようにしてこのファイルを取り込むことも可能です。

```
\documentclass{jarticle}
\input header.tex
\begin{document}
はじめての \TeX ファイルです。
\end{document}
```

2.2.3 dvi ファイルの作成

コマンドプロンプトを起動して上記の作成したファイルのあるフォルダーに移動します。[12] そこで次のコマンドを実行します。

```
platex foo
```

正式には foo.tex ですが拡張子はなければプログラムが補完します (foo というファイルがない場合)。このあと次のようなメッセージが表示されます。

```
This is e-pTeX, Version 3.14159265-p3.7-160201-2.6 (sjis) (TeX Live 2016/W32TeX) (preloaded fo
 restricted \write18 enabled.
entering extended mode
(./foo.tex
pLaTeX2e <2006/11/10> (based on LaTeX2e <2016/03/31> patch level 0)
Babel <3.9q> and hyphenation patterns for 81 language(s) loaded.
(c:/usr/local/share/texmf-dist/tex/platex/base/jarticle.cls
Document Class: jarticle 2006/06/27 v1.6 Standard pLaTeX class
(c:/usr/local/share/texmf-dist/tex/platex/base/jsize10.clo))
No file foo.aux.
[1] (./foo.aux) )
Output written on foo.dvi (1 page, 328 bytes).
Transcript written on foo.log.
(guessed encoding: Shift_JIS = sjis)(guessed encoding: ISO-2022-JP = jis)
```

のようなメッセージが表示されればエラーがなく、foo.dvi ができたと報告されています。

[12] コマンドプロンプトのショートカットを作成して、そのプロパティの作業用のフォルダーを上記の作成したフォルダーにしておくと、毎回、フォルダーへの移動をしなくてすみます。

2.2.4 組版結果の表示

dvi ファイルは組版の結果の情報を保持するものです。この組版の結果を表示する方法としては dvi ファイルをから直接表示する方法と、さらに pdf ファイルに変換して Adobe Reader などの pdf ファイルを表示するツールで表示する方法があります。

dviout でプレビューする

dviout

dviout は日本で開発された dvi ビューワ、印刷ツールです。
出来上がった dvi ファイルを見るためには

```
dviout foo
```

dviout の特徴としては次の点が挙げられます。

- dviout は自分のウィンドウがアクティブになったときに開いている dvi ファイルが更新したかどうかをチェックして、新しいと自動で更新するという機能があります。
- 同じファイルを複数のウィンドウで開くことができます。
- 画像ファイルの内容が画像の内容が中に埋め込まれていません。どこかに持っていくときに忘れる可能性があります。
- 表示できる画像ファイルの形式により扱い方が異なります。
- dviout ではカラーの画像を表示するためにさらなる設定が必
- TeX がインストールしているシステムだけで見ることができます。

Adobe Reader

pdf に変換するればほとんどのプラットフォームで Adobe Reader により pdf ファイルが閲覧できます。メリットとしては次のようなことがあげられます。

- ファイルでは中に画像のデータも含むのでこのファイルだけで完結しています。
- pdf ファイルに「しおり」やハイパーリンクを付けることで電子文書の利点を活用することができます。
- Adobe Reader で開いているファイルの上書きができません。したがって、修正するごとに、Adobe Reader を終了してから pdf ファイルを作成することになります。
- 画像ファイルとしていくつかの種類の画像フォーマットが扱えます。

とコマンドプロンプトで打ち込みます。dviout はフォーカスが与えられたときに dvi が以前のものと違っているかを判定して最新のものを表示しますので一度起動したら終了させる必要はありません。

pdf ファイルを作成する

dvi ファイルを pdf ファイルに変換する方法は次の方法があります。

dvipsfm(x) を用いる LaTeX の配布パッケージには dvi ファイルを pdf ファイルに変換する dvipdfm というプログラムがついています。

```
dvipdfm foo
```

と入力すると

```
foo -> foo.pdf
[1]
6196 bytes written
```

などと表示されて pdf ファイルが作成されます。エクスプローラーからファイルをダブルクリックすれば Adobe Reader で表示できます。

なお、Adobe Reader は表示しているファイルを占有しています。したがって、表示したままで同じ名前の pdf ファイルを作成しようとすると

```
foo -> foo.pdf

xdvipdfmx:fatal: Unable to open "foo.pdf".

Output file removed.
```

というメッセージが出て更新に失敗します。

なお、hyperref パッケージを利用する場合には少し注意が必要な場合があります。

gswin32c を用いる dvi ファイルをいったん、PostScript ファイルに変換し、gswinXXc の機能を用いて pdf ファイルに変換する方法があります。この方法は複雑なので第 9 章で解説します。

2.2.5 エラーへの対処

LaTeX は目次を作成したり相互参照をする機能が備わっています。これらの情報は同じファイルを LaTeX にかける直前に得られたものを用いて行われます。したがって、初めて LaTeX かけたときには目次の表題は現れますが中身はありません。正しい目次を得るためにはもう一度 LaTeX にかけてください [13]。相互参照などが変化した場合には LaTeX がもう一度 LaTeX を通すようにというメッセージが出ます。

LaTeX で文書を作成するとエラーが発生して dvi ファイルが作成できなくて途中で ? がでて止まる場合があります。この場合には x を入力すると処理を中断できます。

いくつか代表的なエラーを挙げておきます。

[13] この考え方は計算機のプログラミングの初期にあった 2 パスアッセンブラと呼ばれる技法と同じものです。

- `! Undefined control sequence.`
  LaTeX のコマンドが定義されていないという意味です。スペルの間違いかコマンド名の後に適切な区切りの文字がない場合が多いです。次の行にエラーが発生した場所が行番号とともに示されます。

- `Runaway argument?`
  マクロに渡した引数の始まりを示す { に対応する } がない場合に現れます。示された位置以降に } を入れましょう。

- `! Missing $ inserted.`
  数式モードでしか使えないマクロが現れた場合におきます。与えられた部分を数式モードにするように設定します。また、数式モードの中で段落があってはいけないもの (一行表示の `\[\]` や `equation` 環境内など) 見やすくするために空行を入れたければ行頭にコメントを示す%を入れてください。

- `! LaTeX Error: There's no line here to end.`
  強制改行 (`\\`) をしようとしたときにすでに行が改行されていて中身がない場合に発生します。`\␣`(␣は半角の空白) などを追加すると消えます。環境が終了した直後などでも起きるものがあります。

- ```
  ! LaTeX Error: Command \TeX already defined.
                 Or name \end... illegal, see p.192 of the manual.

  See the LaTeX manual or LaTeX Companion for explanation.
  Type  H <return>  for immediate help.
  ...
  ```
  すでに定義されているコマンド (マクロ) を定義しようとしたというエラーです。既存のコマンドを再定義する場合には `\renewcommand` を用います。

このほかに警告文法上の間違いではないが問題がある警告もあります。

- `Overfull \hbox (115.42491pt too wide) in paragraph at lines 156--156`
  `\hbox` は水平方向の箱 (horizontal box) の意味です。水平方向に配置がはみ出したことを意味しています。この値が小さければ見た目にそれほど不自然なことはありません (ここでの値は大きすぎます)。なお、縦方向の場合には `vbox` というメッセージになります。

- `Underfull \hbox (badness 3078) in paragraph at lines 67--71`
  水平方向に配置するものが足りなかった (`Underfull`) ことを警告しています。この部分では空白が多くなっているはずです。

- ```
  LaTeX Font Warning: Font shape `JY1/mc/m/sc' undefined
  (Font)              using `JY1/mc/m/n' instead on input line 154.
  ```
  LaTeX では同一のフォントに対し形を変えたものを用います。アルファベットではそのヴァリエーションが用意されているのですが日本語のフォントでは用意されていないので標準のものを使用するというメッセージです。無視してかまいません。

なお、最後でも `LaTeX Font Warning: Some font shapes were not available, defaults substituted.` という警告がでます。

第3章　LaTeX ファイルを作る

3.1　LaTeX ファイルの構造

LaTeX のファイルは次のような構造をとります。

```
1 \documentclass[a4j]{jarticle}
2
3 <プリアンブル>
4
5 \begin{document}
6 <本文を書く>
7 \end{document}
```

一番初めにある \documentclass はこれから作成する文書の全体的な構成 を定義します (利用する文書クラスの宣言)。ここでは jarticle を利用します。このほかに代表的なものは jreport や jbook があります。このほかに代表的なものは jreport や jbook があります。この指定によっては文書を構成するコマンド (章、節、項など) が違ったりします。

この前にある [と] で挟まれた部分は jarticle に対するオプションです。一般に LaTeX のコマンドの後のあるる [と] で挟まれた部分は常にオプションです。なくてもかまわない場合があります。ここでは a4j なので日本語の a4 版を想定した印刷範囲の幅や高さ余白が設定されることになります。この値が気に食わないときには設定を変更することも可能です。なお、用紙の種類としては b5j、b4j などがあります。

デフォルトのフォントの大きさは 10pt です。フォントの大きさで用意されているオプションは 11pt と 12pt です。

また、プレゼンテーションをする場合には用紙を横向きにますがこの設定は landscape というオプションを指定します。

これらのオプションを浮く数指定するときはコンマで区切って記述します。

\begin{document} までの部分はプリアンブルと呼ばれ、実際に印刷される文書の範囲ではありません。印刷部分の大きさを変更するような記述は原則としてここに書くことになります。

\begin{document} と \end{document} にはさまれた部分が実際の本文になります。\end{document} 以降はまったく印刷の対象になりません。

3.2　LaTeX の約束事

LaTeX(TeX) で文書を記述する上での約束事について順に説明します。

3.2.1 特別な扱いの文字

次の文字は LaTeX では元来の文字の意味を持ちません。LaTeX では別の意味で用いられます。

\ % { } $ ^ _ &

**** \ で始まる文字列は LaTeX のコマンド (マクロ) です。名称の切れ目は括弧や数字、空白、\ です（つまり、これらの文字はマクロの名称には含むことが通常できません)。ただし、全角の数字や空白は区切りとならないので注意しましょう。

\ を書きたい場合には \textbackslash と書きます。なお、\\ と書くとそこで改行されます。他の文字も同様です。

% %以降行末 (改行文字を含む) まではコメントと見なされ、印刷には現れません。

{ と } { と } で囲むとその範囲がグループ化されます。グループ化するとその部分はひとつのものとして扱われます。一般にグループ内で変更した各種の設定はグループから出た後では元に戻ります。

- ある部分のフォントの大きさなどを一時的に変える。
  フォントの大きさはグループの外では元に戻ります。
- マクロに渡す引数の区切りとして使用する。

{ や } を表したいときにはそれぞれ \{ と \} と書きます。

**$, _, ^** LaTeX には通常の文章を組むモードと数式を組むための 2 つのモードがあります。二つの$で挟まれた部分は文章内にある数式モードで組版されます。_と ^は数式を表すときに利用します。これらの文字を表示するときには $ は \$ または \textdollor、_と ^はそれぞれ \textunderscore と \textasciicircum と書きます。

3.2.2 空白と段落

空白はいくつあってもひとつと見なされます。改行は空白と同じ役割をします。また、行の先頭の空白は無視されます。なお、漢字の空白 (いわゆる全角スペース) は空白とはみなされません。

何も記述がない行があるとそこが段落の始まりと見なされます。段落の始まりには指定された量の空白が入ります。この空白を入れたくない場合には \noindent をいれます。

改行は LaTeX がアルゴリズムにしたがって行います。気に食わないときには \\ を用いて強制的に変えることも可能です。なお、\\ のあとに [1cm] などと書くことで改行幅の調節ができます。逆にこのことが悪さをしては \\ の次に [を出力しようとすると「長さが指定されていない」というエラーがでます。そのような場合には\\{}のように空のグループを作るとよいでしょう。

LaTeX のロゴは \LaTeX と書きます。この後に必ず半角の空白を入れてください。一般にはこのロゴの後には空白が入りません。強制的に空白を入れたい場合には \␣(␣は空白を意味します) と書いてください。

やってみよう 3.1 簡単な文書を作成して次の事柄を確認しなさい。

1. `\documentclass` を `jarticle` から `jreport` に変える。
2. オプションの `a4j` を `b5j` に変える。
3. オプションに `11pt` を追加する。
4. オプションに `12pt` に変える。
5. 段落を入れる。
6. %の文字を 10 個連続して出力する。
7. { を出力する。
8. \ を出力する。
9. LaTeXLaTeX LaTeX LaTeX と出力してください (後のほうが空白が広くなっています)。
10. 作成した `.tex` ファイルのほかにどのようなファイルがカレントディレクトリのできているか調べなさい。

3.3 フォントを変える

LaTeX でフォントの大きさは数値ではなく記号で指定します。また、先に述べた文書の標準のフォントの大きさでもフォントの大きさは変わります。次の記号が定義されています。`{\large aaa}` のように利用します。

指定の方法	10pt の時の大きさ	例
`\tiny`	5pt	ABCabc あいうえお
`\scriptsize`	7pt	ABCabc あいうえお
`\footnotesize`	7.5pt	ABCabc あいうえお
`\small`	8pt	ABCabc あいうえお
`\normalsize`	10pt	ABCabc あいうえお
`\large`	12pt	ABCabc あいうえお
`\Large`	14pt	ABCabc あいうえお
`\LARGE`	17pt	ABCabc あいうえお
`\huge`	20pt	ABCabc あいうえお
`\Huge`	25pt	ABCabc あいうえお

フォントの形状を変化させるコマンドには 大きさを変えるものと同じ形の利用するものとコマンド型のものの二通りがあります。

コマンド型	宣言型	例
\textrm{...}	{\rmfamily...}	ABCabc あいうえお
\textsf{...}	{\sffamily...}	ABCabc あいうえお
\texttt{...}	{\ttfamily...}	ABCabc あいうえお
\textmd{...}	{\mdsereis...}	ABCabc あいうえお
\textbf{...}	{\bfsereis...}	**ABCabc あいうえお**
\textup{...}	{\upshape...}	ABCabc あいうえお
\textit{...}	{\itshape...}	*ABCabc* あいうえお
\textsl{...}	{\slshape...}	*ABCabc* あいうえお
\textsc{...}	{\scshape...}	ABCABC あいうえお
\emph{...}	{\em...}	*ABCabc* **あいうえお**
\textnormal{...}	{\normalfont...}	ABCabc あいうえお

これらのフォントの形状と大きさを変えるコマンドは組み合わせて使えます。

やってみよう 3.2 いろいろなフォントの大きさを変えたり、形状を変えてみよう。

3.4 原稿を見る

次のような原稿を考えて見ましょう。

```
1  \documentclass[a4j]{jarticle}
2  \usepackage{fancyvrb}
3  \newtheorem{Try}{やってみよう}[chapter]
4  \title{\LaTeX の初歩}
5  \author{平野 照比古}
6  \date{}
7  \begin{document}
8  \maketitle
9  \tableofcontents
10 \newpage
11 \section{\LaTeX ファイルを作る}
12 \section{\LaTeX ファイルの構造}
13 \LaTeX のファイルは次のような構造をとります。
```

- 1 行目は \documentclass の宣言です。ここでは 25 ページと同様に jarticle にしています。オプションとして a4j を指定しています。
- 2 行目はプログラムなどのリストに自動で番号を振るための環境 (後述) が定義されているファイル (fancyvrb) を読み込むことを宣言しています。LaTeX ではこのようにいろいろな目的のために開発されたパッケージが存在します。これをうまく使うことでいろいろな文書が比較的簡単に作成できることになります。

- 3 行目は新しい定理環境を定義しています (\newtheorem)。この文書で「やってみよう」と書かれた場所がいくつか登場していますが、この表現ができるようにするための宣言です。Try と書いている部分がこの環境の定義になります。この先、\begin{Try} と \end{Try} でこの環境を使用できます。新しい環境には通し番号が自動で付きます。この例ではこの後に [chapter] が付いていますが、これは chapter(章) が変わるごとにこの番号がリセットされるように指示しています。この場合ですと「章番号. 通し番号」の形になります。[1]
- 4 行目はこの文書のタイトルを定義するマクロです。この場所でタイトルが出力されるわけではありません。
- 5 行目は著者を定義しています。
- 6 行目は日付を定義しています。中身がないので日付が出力されません。
- 7 行目から本文が始まることを宣言しています。
- 8 行目の \maketitle でここにタイトルを表示することを指示しています。\maketitle は 1 回しか有効ではありません。
- 9 行目は目次の出力 (\tableofcontents) を指示しています。
- 10 行目は目次の後で強制的に改ページすることを指示しています。
- 11 行目は新しい節の始まりを示しています。{} 内にその節のタイトルを書きます。
- 12 行目は項の始まりを示しています。文書の構造を決めるものとしてはこのほかに \subsubsection, \paragraph があります。
- \documentclass jreport には chapter が、jbook には chapter と part があります。

やってみよう 3.3 次のことをして結果がどうなるか調べなさい。

1. 2 行目の \usepackage の行をコメントアウトする。
2. \documentclass を jreport にする。
3. 3 行目の \newtheorem の定義で [chapter] を省略した場合
4. 6 行目の \date の {} 内に文字を書く。
5. 6 行目の \date を省略する。
6. \title, \author, \date の宣言順序を変えた場合のタイトルの表示が変わるかどうか確認する。
7. \maketitle を 2 回使用しても 2 回目には出力がないことを確認する (エラーにならない)。
8. 作成した.tex ファイルを platex に通したときには目次の内容が出力されているか。

[1] 正確には \documentclass が jbook の場合です。

3.5 いろいろな文書の形式

いろいろな環境を利用することで文書に変化をつけることができます。これらの環境の前後では一般に行間が空きます。

3.5.1 箇条書きの書き方

箇条書きには項目の先頭に • がつく itemize 環境と番号がつく enumerate 環境があります。これらの環境を利用するためには \begin{itemize} と \end{itemize} (または \begin{enumerate} と \end{enumerate}) ではさみます。各項目の先頭には \item をつけて項目の開始を指示します。

```
\begin{itemize}
 \item 箇条書きの先頭はこのようになります。
 \item[タイトル]
    タイトルをつけることもできます。
\end{itemize}
```

- 箇条書きの先頭はこのようになります。

タイトル タイトルをつけることもできます。

タイトルの右端が項目の先頭マークの位置にそろえられるのであまり長いとはみ出しすぎることに注意しましょう。

番号付は次のようになります。

```
\begin{enumerate}
 \item 番号付箇条書きの先頭はこのようになります。
 \item 番号は自動でつきます。
 \item[タイトル] タイトルをつけることもできます。
\end{enumerate}
```

1. 番号付箇条書きの先頭はこのようになります。
2. 番号は自動でつきます。

タイトル タイトルをつけることもできます。

これらのふたつの環境の中にさらに同様の環境を入れることもできます。そのときは番号や先頭につく記号が換わります。

やってみよう 3.4 箇条書きを書いて見ましょう。また、箇条書きの中に箇条書きを入れ込んだ場合にどのようになるか確かめなさい。

やってみよう 3.5 **やってみよう** *3.3*の問題文の箇条書きの部分を書いてみましょう。

3.5.2 中央ぞろえ

文を中央にそろえたいときには center 環境を用います。複数の行を連続して中央ぞろえにしたい場合には切れ目に\\を入れます。

```
\begin{center}
 この文は中央にそろえられます。\\
 改行には\textbackslash\textbackslash
 を用います。\\
\LARGE 文字を大きくします。\\
元の大きさに戻りません。
\end{center}
```

この文は中央にそろえられます。
改行には \\ を用います。
文字を大きくします。
元の大きさに戻りません。

やってみよう 3.6 表題と同じ形式の出力を center を用いて作成しなさい。

3.5.3 右ぞろえ

右揃えにするための環境は flushiright 環境です。

```
\begin{flushright}
この行は右寄せになります。\\
\verb+center+環境と同様に改行は
\textbackslash\textbackslash です。\\
ここに今日の日付を入れました。\today
\end{flushright}
```

この行は右寄せになります。
center 環境と同様に改行は \\ です。
ここに今日の日付を入れました。平成 29 年 10 月 3 日

ここでは今日の日付を表示する \today を用いています。

3.5.4 脚注

脚注を入れる場合には脚注の番号が入る位置に \footnote コマンドを入れます。

```
脚注を入れるためには
\texttt{\textbackslash footnote}
コマンドを用います。
\footnote{脚注は普通ページの下部に
入りますがここではページ内にページを
作成する\texttt{minipage}環境を
使用しているのでこの部分に入ります。}
```

脚注を入れるためには \footnote コマンドを用います。[a]

[a] 脚注は普通ページの下部に入りますがここではページ内にページを作成する minipage 環境を使用しているのでこの部分に入ります。

3.5.5 ページ内にページを作る

上の表示にもありますがページ内に小さなページを作成する minipage 環境というものがあります。ここでは minipage を二つ並べて左に LaTeX のソースコードを、右にその結果を表示するという技法を使っています。minipage 環境は次のような形で使用します。

```
\begin{minipage}[t]{10cm}
本文をかく
\end{minipage}
```

minipage のあとの [t] はオプションのパラメータで、表示する位置を示します (この場合は t なので上にそろいます)。次の引数は minipage の幅を指定します。ここでは 10cm の幅となります。

3.5.6 プログラムリストを表示する

プログラムなど計算機関係のファイルを表示するときは等間隔のフォントを用いることが多くあります。これを実現する環境が fancyvrb パッケージにある Verbatim 環境です [2]。

```
\begin{Verbatim}
#include<stdio.h>

int main( void ) {
  printf("Hello World\n");
  return 0;
}
\end{Verbatim}
```

```
#include<stdio.h>

int main( void ) {
  printf("Hello World\n");
  return 0;
}
```

リストに対して番号を振るためには Verbatim 環境にオプションの引数 [numbers=left] を追加します。

\begin{Verbatim}[numbers=left, firstnumber={1 行目の行番号 },stepnumber=数字]
...\end{Verbatim}

stepnumber の数ごとに行番号が出力されます。

```
\begin{Verbatim}[numbers=left,stepnumber=2]
#include<stdio.h>

 main( void ) {
  printf("Hello World\n");
  return 0;
}
\end{Verbatim}
```

```
#include<stdio.h>
2
 main( void ) {
4  printf("Hello World\n");
   return 0;
6 }
```

なお、Verbatim のオプション firstnumber には数のほかに last を指定できます。この場合には直前の Verbatim の番号を引き継ぎます。

```
\begin{Verbatim}[
  numbers=left, firstnumber=last]
int foo(int x, int y ) {
   return x+y;
}
\end{Verbatim}
```

```
7 int foo(int x, int y ) {
8    return x+y;
9 }
```

[2] 以前は verbatim 環境や moreverb パッケージの listing 環境を使用していましたがこちらの方がいろいろなことができるので 2016 年度から変更しました。

上の例では直前の Verbatim 環境が 6 行目で終わっているのでここでは行番号が 7 から始まることになります[3]。

プログラムリストなどでは空白を明示的に表したい場合があります。その場合には Verbatim*環境を用います。空白は␣と表されます。

```
\begin{Verbatim*}
#include<stdio.h>

inat main( void ) {
  printf("Hello World\n");
  return 0;
}
\end{Verbatim*}
```

```
#include<stdio.h>

inat␣main(␣void␣)␣{
␣␣printf("Hello␣World\n");
␣␣return␣0;
}
```

この資料では他のファイルを Verbatim 環境の中に取り入れる \VerbatimInput を利用しています (69 ページ参照)。なお、\VerbatimInput は \VerbatimInput[オプション]{ ファイル名 } のかたちで利用します。

リストの表示についてはいろいろカスタマイズができます。次の例はリスト全体を囲む枠を作り、上下にタイトルをつけています。

```
\begin{Verbatim}[numbers=left,
  frame=single, framesep=3mm,
  label={[hello.c--top]hello.c--end},
  labelposition=all]
#include<stdio.h>

int main( void ) {
  printf("Hello World\n");
  return 0;
}
\end{Verbatim}
```

hello.c--top

```
1 #include<stdio.h>
2
3 int main( void ) {
4   printf("Hello World\n");
5   return 0;
6 }
```

hello.c--end

- frame で枠の形式を指定します。ここでは single(全体を囲む) を指定しています。ほかには none(枠なし)、leftline(左側)、topline(上)、bottomline(下) と lines(上下) が指定できます。
- 枠の太さは framerule で、rulecolor で枠の色、fillcolor でテキストと枠の間の色を指定します。
- framesep はリストの表示域と枠との距離を指定します。
- labelposition は表題を表示する位置を指定します。none(表題なし)、topline(上)、bottomline(下) と all(上下) が指定できます。

[3]原稿を作成している段階で 2 つおきに番号をつけていたとき、最後の番号が表示されていないと引き継いだ番号が大きくなっているというバグがありました。

- label は枠の上下に表示するテキストを定義します。上下に表示するときに、上限で異なる表題にしたいときには例のように label の指定時に [] 内のテキストが上部の表題に使われます。

行番号はカウンター FancyVerbLine で管理されているので、この値を表示するマクロ\theFancyVerbLine を再定義すれば表示を変えられます。

やってみよう 3.7 自分の書いたプログラムなどを Verbatim 環境の中に取り入れてみましょう。

3.5.7 強制改ページ

強制改ページをするためには通常は \newpage コマンドを用います。このコマンドを用いても後で解説する、表 (table 環境) や図 (figure 環境) が思った位置に出ない場合があります。このようなときには \clearpage を用いるとよいでしょう。なお、次のページを奇数にする \cleardoublepage というものもあります。

やってみよう 3.8 \newpage を 2 回続けて書いた場合どのようになるか?

3.5.8 番号が自動でつく環境を定義する

このテキストでは **やってみよう** という環境が定義されていてそれに通し番号がついています。このような環境は \newtheorem を用いて作成しています。ここでは次のように定義しています。

```
\newtheorem{Try}{やってみよう}[chapter]
```

newtheorem の後に続く{Try}は新しい環境の名称 Try を与えます。その後に続く [chapter] の部分は [] なのでオプションの引数です。ここでは chapter が指定されているので番号付けが章 (chapter) が変わるごとにリセットするように指示しています。この定義により

```
\begin{Try}
なにかを書く。
\end{Try}
```

として利用できます。また、

```
\begin{Try}[表題つき]
表題もつきます。
\end{Try}
```

とすると次の「やってみよう」のように表題を付けることもできます (ここでは [番号が変わります] としました)。

やってみよう 3.9 (番号が変わります) 次の定義の場合番号がどのようになるか確かめなさい。また、オプションの引数がない場合にはどのようになるか確かめなさい。

```
\newtheorem{TryB}{やってみよう}[section]
\begin{TryB}
 番号はどうなるかな？
\end{TryB}
```

3.6 その他知っておくと便利なこと

3.6.1 番号付け

LaTeX では番号を管理するためにカウンターという機能を用いています。カウンターには整数を格納できます。ページを記録する page カウンター、章の番号を記録する chapter カウンターなどがあります。\newtheorem で新しい環境を定義すると自動的にカウンターがひとつ割り当てられます (LaTeX ではカウンターの数は全体で 255 個しかありません) [4]。カウンターの値を表示するためには通常 \the<カウンター名>というマクロを利用します。ページ番号であれば \thepage です。このテキストでは目次までのページではローマ数字で、本文では一度ページ番号を元に戻してアラビア数字に直しています。これは文書のはじめに次のように記述しているからです。

```
1 \begin{document}
2 \maketitle\clearpage
3 \input copyright.tex
4 \renewcommand{\thepage}{\roman{page}}
5 \tableofcontents
6 \cleardoublepage
7 \setcounter{page}{1}
8 \renewcommand{\thepage}{\arabic{page}}
```

- 4 行目で \thepage を \roman{page} に再定義 (\renewcommand) しています。
- 本文が開始するときにページの番号を 1 に戻します (7 行目の \setcounter{page}{1})。
- また、8 行目で \arabic{page} として算用数字で表示するように再定義しています。

カウンターの値を変化させる 7 行目ではカウンターの値を指定した値に設定していますが、\addtocounter を用いるとカウンターの値を指定した値だけ変化させられます。

```
\addtocounter{page}{2}
```

なお、あるカウンターの値を別のカウンターの値だけ加えるためには

```
\addtocounter{distcnt}{\value{refcnt}}
```

のように参照するカウンターの前に \value を付けます。

[4] ほかの環境などと共有することも可能です。

新しいカウンターを作る 新しいカウンターを作るためには \newcounter を用います。

```
\newcounter{newname}[oldcounter]
```

oldcounter の部分はオプションです。ある場合には oldcounter の値が変化すると newname の値が 0 になります。

カウンターの値の表示を変える 次にカウンターの表示はアラビア数字やローマ数字だけではありません。次のようなものも使えます。

アラビア数字	\arabic{chapter}	3
小文字のローマ数字	\roman{chapter}	iii
大文字のローマ数字	\Roman{chapter}	III
小文字のアルファベット	\alph{chapter}	c
大文字のアルファベット	\Alph{chapter}	C
脚注用の文字	\fnsymbol{footnote}	§

なお、このテキストでは節 (section) の番号は「章の番号. 節の番号」の形をしていますがこれは \thesection を \thechapter.\arabic{section} と定義しているからです。この \thesection は相互参照の時に使われる数字になりますのでこの文字を大きくしたりすると参照時の文字列も大ききなってしまいます。これを避ける方法は 8.1.3 を参照してください。

番号付きの箇条書きに対してもそれぞれカウンターが定義されていて (一番上のカウンター名は \enumi、以下順に \enumii、\enumiii、\enumiv まであります) その定義を直すことで番号付けの方法を変えることができます。

3.6.2 入力ファイルの分割

すてにプログラムのリストを表示するためにそのファイルを取り込む方法を示しました。ここでは文書全体を分割して管理するための方法を示します。

ある程度文書の量が多くなるとひとつのファイルでは管理が大変になります。LaTeX では他のファイルを取り込む \input があります。\input foo.tex で foo.tex が取り込まれます。このテキストは章ごとにファイルを分割しています。

このほかに \include があります。これは \input と同じように指定されたファイルを取り込みますが、\includeonly というコマンドがあるとこれに指定されているファイルだけを読み込むという機能が追加されています。非常に大きな文書を作成するときに、すべてをやり直すと非常に時間がかかるので、一部部分訂正したときの確認ようにあるものです。最近のパソコンは十分速く、筆者の経験でも数百ページの文書でもそれほど時間はかからないのでこのマクロを使う場面は限られてくると思います。用途からも分かるように \includeonly で読み込まれたファイルの中で \includeonly は使えません。

3.6.3 まとまった部分をコメントにする

%を用いると行末までコメントになることはすでに 26 ページで述べました。ある部分をまとめてコメントアウトするために% をたくさん行頭に付けるのは面倒です。コメントアウトする範囲を \iffalse と \fi ではさむとその部分はコメントアウトされます。

3.6.4 印刷範囲を大きくする

与えられた紙の大きさは \documentclass のオプションで [a4j] や [b5j] などと定義します。このオプションを指定すると自動的に上下左右のの空白が決まります。印刷される範囲は次のディメンションレジスタ (長さを閉まっておく場所) で決まります。

\hoffset	紙の上端からの位置。この値に 1 インチ加えたものが実際の値。
\voffset	紙の左端からの位置。この値に 1 インチ加えたものが実際の値。
\topmargin	\voffset の位置とヘッダーの上部との間隔
\headheight	ヘッダーの高さ
\headsep	ヘッダーの下端とテキストの間隔
\oddsidemargin	\hoffset との位置と印刷部分の間隔
\textwidth	印刷部分の幅
\textheight	印刷部分の高さ

実際の印字位置は上から

1 インチ + \voffset + \topmargin + \headheight + \headsep

に、左から

1 インチ + \hoffset + \oddsidemargin

の位置から印刷されます。

3.6.5 均等割付

ある範囲に文字を等間隔に並べたい場合があります。TEX には無限に伸び縮みができる空白があります。これは無限に伸びるばねのと考えるとわかりやすいでしょう。入れられた場所の前後にある文字 (ボックス) を押して文字の位置を変化させます。この単位は \fill です。水平方向の空白を入れるマクロは \hspace です。このコマンドの場合、行の最後にある場合には無視されます。なお、\hspace{\fill} の簡略形として \hfill があります。この空白を必ず入れたい場合には \hspace* を用います。次の例で見比べて見ましょう。

```
\fbox{\parbox[t]{\textwidth}{%
\hspace{\fill}aaa bbb\hspace{\fill}\\
\hspace*{\fill}aaa bbb\hspace{\fill}\\
\hspace{\fill}aaa bbb\hspace*{\fill}\\
\hspace*{\fill}aaa bbb\hspace*{\fill}\\
\hfill aaa bbb\hfill}}
```

aaa bbb
aaa bbb
aaa bbb
aaa bbb
aaa bbb

- この例では印字範囲がどこかわかるように全体を枠で囲っています。そのマクロが \fbox です (1 行目)。
- 幅を指定した段落を作るために \parbox(paragraph box) を利用しています。次のオプションの引数 [t] はこのボックスを上端位置にそろえることを指定しています。この引数の意味は tabular 環境と同じです。次の引数でこのボックスの幅を指定します。\textwidth が指定されています。
- 1,2,5 行目では右に書いてある空白が意味を成していないことが表示からわかります。

ある範囲に均等割付をするためにはこの \hspace{\fill} を文字の間に入れればできることになります。次の例は 5cm の幅の間に均等割付をしています。最後は \hfill と書けないことに注意してください。

```
\framebox[5cm]{
  \hfill 出\hfill 発\hfill 日\hspace*{\fill}}
```

出　　発　　日

ここでは枠付きのボックスを \framebox で作成しています。このマクロはオプションの引数でその幅を指定できます。

与えられた文字数に応じて毎回調整するのが面倒であればマクロを組むことになります。文字数ごとに作るのであれば簡単ですが、文字数にかかわらずひとつのマクロを組むのは少し面倒です。詳しくは第 7 章を見てください。

やってみよう 3.10 上記の例で枠の幅は本当に 5cm になっているか確認しなさい。

\fill と同じようなものとして \stretch があります。これは数字の引数をとります。\fill と \stretch{1} は同じものです。この値を変えると対応する空白の割合が変化します。

```
\framebox[5cm]{
 \hspace{\stretch{2}}出\hfill 発\hfill 日
 \hspace*{\stretch{2}}}
```

出　発　日

この例では両端の値を 2 としていますので中央よりの空白の 2 倍が両端の空白になります。

なお、\hspace の代わりに \vspace を用いると縦方向の制御ができます。このテキストの表紙の次のページでは\vspace{\fill}を用いて出力を下に詰めています。これがそのリストです。

```
\clearpage
\vspace*{\fill}
```

この文書についてお問い合わせは下記のメールまでご連絡ください。

```
\begin{flushright}
 \parbox{20em}{\copyright 2006/9-2017/10 平野照比古\\
\texttt{e-mail:hilano@ic.kanagawa-it.ac.jp}\\
\texttt{URL:http://www.hilano.org/hilano-lab}}
\end{flushright}
```

やってみよう 3.11 ページの上下方向の中央にテキストを配置するためにはどうすればよいか考えなさい。

3.6.6 アクセントのついた文字

LaTeX ではキーボードにない文字はマクロで指定します。たとえばフランス語やドイツ語などで使われるアクセントのついた文字もそのひとつです。

`\"e`	ë	`\"a`	ä	`\"A`	Ä
`\'e`	é	`\'a`	á	`\'A`	Á
`\^e`	ê	`\^a`	â	`\^A`	Â
`\`e`	è	`\`a`	à	`\`A`	À

3.6.7 特殊な文字

¥のような通貨の単位は全角の文字を用いなければ特別な形の記述をします。なお、これらの文字を使いたい場合には textcomp パッケージを利用してください。

¥	`\textyen`	€	`\texteuro`	£	`\pounds`
$	`\textdollar`	¢	`\textcent`		
©	`\copyright`	®	`\textregistered`	™	`\texttrademark`
§	`\textsection`	¶	`\textparagraph`	‰	`\textperthousand`

第4章　表組み

4.1　表組みの基本

表を組むための環境は tabular です。\begin{tabular} の後に各列の表示方法を指定する文字列を { と } の間に書きます。なお、\begin{tabular} の直後にオプションのパラメータを入れることができます。これは tabular 環境で構成されたものの基準線 (ベースライン) の配置位置を指定するものです。

- [t] 基準線が一行目の基準線の位置のになります。
- [b] 基準線が最下位行の基準線の位置になります。
- [c] または指定しない場合 全体の中央位置になります。

基本的な列記述子　項目を列に配置するための方法は次の例を見てください。

```
\begin{tabular}[t]{|c|c|}
\hline
記号&意味\\\hline\hline
\texttt{c}&中央ぞろえ\\\hline
\texttt{r}&右寄せ\\\hline
\texttt{l}&左寄せ\\\hline
\texttt{|}&縦の罫線を入れる\\\hline
\texttt{p}&指定した幅の段落を入れる\\\hline
\texttt{*}&フォーマットの繰り返し\\\hline
\end{tabular}
```

記号	意味
c	中央ぞろえ
r	右寄せ
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

- 項目の区切りは&で与えます。なお、& を書きたい場合には \& と書きます。
- \hline はこれから開始する行の上に横幅いっぱいの区切り線を書くことを指示します。
- \\ は一行の終わりを示します。
- \begin{tabular} の後にある [t] なので配置位置は第 1 行目となります。
- 各列の幅はは一番幅の大きい項に対して設定されます。列の間には適当な空白が入ります (調整することも可能です)。

p 記述子　p は列の幅を指定します。p{3cm} とすれば幅が 3cm になります。

```
\begin{tabular}[t]{|c|p{3cm}|}
\hline
記号&意味\\\hline\hline
\texttt{c}&中央ぞろえ\\\hline
\texttt{r}&右寄せ\\\hline
\texttt{l}&左寄せ\\\hline
\texttt{|}&縦の罫線を入れる\\\hline
\texttt{p}&指定した幅の段落を入れる\\\hline
\texttt{*}&フォーマットの繰り返し\\\hline
\end{tabular}
```

記号	意味
c	中央ぞろえ
r	右寄せ
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

p で配置された項目の上下の位置は隣の項目の基準線と同じ 1 行目になっています。

この配置を中央でそろえるためには tabular の中に tabular を入れることで解決できます。

前の内容と同じものを表してみます。内部の tabular で表された位置に注意してください。ベースラインの位置を指定していないので中央が設定されています。

また、空白の大きさが以前のものと違うことも確認しておいてください。

```
\begin{tabular}[t]{|c|c|}
\hline
記号&意味\\\hline\hline
\texttt{l}&\begin{tabular}{p{3cm}}
            左寄せ
            \end{tabular}\\\hline
\texttt{|}&\begin{tabular}{p{3cm}}
            縦の罫線を入れる
            \end{tabular}\\\hline
\texttt{p}&\begin{tabular}{p{3cm}}
            指定した幅の段落を入れる
            \end{tabular}\\\hline
\texttt{*}&\begin{tabular}{p{3cm}}
            フォーマットの繰り返し
            \end{tabular}\\\hline
\end{tabular}
```

記号	意味
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

内部の table 環境でも列間の空白が入るので前の例よりも縦線の間が空きます。

@記述子　次の例では内側にある列間の空白を取り除くために@{} を用いています。

```
\begin{tabular}[t]{|c|c|}
\hline
記号&意味\\\hline\hline
\texttt{l}&\begin{tabular}{@{}p{3cm}@{}}
            左寄せ
            \end{tabular}\\\hline
\texttt{|}& \begin{tabular}{@{}p{3cm}@{}}
            縦の罫線を入れる
            \end{tabular}\\\hline
\texttt{p}&\begin{tabular}{@{}p{3cm}@{}}
            指定した幅の段落を入れる
            \end{tabular}\\\hline
\texttt{*}&\begin{tabular}{@{}p{3cm}@{}}
            フォーマットの繰り返し
            \end{tabular}\\\hline
\end{tabular}
```

記号	意味
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

元来、@{} は列間の空白の代わりに入れるものを定義するためのものです。内側の tabular 環境で列間の空白を取り除いているので前の例よりすっきりした表になっています。45 ページにも@ 記述子の例がありますので参考にしてください。

なお、4.2 で解説している array パッケージを用いると tabular 環境を用いなくても同様のことが可能になります。

***記述子** 次の表は九九の表です。同じフォーマットで列を構成したいので同じ形式の列の表示繰り返す*を用いて列の構成をしています。

```
\begin{tabular}[t]{|c||*{9}{r|}}
\hline
  &  1&  2&  3&  4&  5&  6&  7&  8&  9\\\hline\hline
 1&  1&  2&  3&  4&  5&  6&  7&  8&  9\\\hline
 2&  2&  4&  6&  8& 10& 12& 14& 16& 18\\\hline
 3&  3&  6&  9& 12& 15& 18& 21& 24& 27\\\hline
 4&  4&  8& 12& 16& 20& 24& 28& 32& 36\\\hline
 5&  5& 10& 15& 20& 25& 30& 35& 40& 45\\\hline
 6&  6& 12& 18& 24& 30& 36& 42& 48& 54\\\hline
 7&  7& 14& 21& 28& 35& 42& 49& 56& 63\\\hline
 8&  8& 16& 24& 32& 40& 48& 56& 64& 72\\\hline
 9&  9& 18& 27& 36& 45& 54& 63& 72& 81\\\hline
\end{tabular}
```

	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

4.2 array パッケージ

array パッケージ (実はこのテキストも利用しています) は LaTeX で定義している列の配置を一部変え、いくつかの記述のパラメータを追加しています。

記述子	変更	内　　容
|	変更	縦線を入れる。二つの列の間がこの線の幅だけ増える
m{幅}	追加	列幅が指定された値で上下の配置が中央ぞろえになる
b{幅}	追加	列幅が指定された値で上下の配置が下の位置ぞろえになる
>{宣言}	追加	r,l,c, p{...},m{...},b{...} の直前に使用可能。「宣言」の部分が各列の前に入る。
<{宣言}	追加	r,l,c, p...,m...,b... の直後に使用可能。「宣言」の部分が各列の後に入る。
!{宣言}	追加	@{...}と同じだけれど列間の空白は所定の値が入る

以前の例を書き直して見ます。

```
\begin{tabular}[t]{|c|m{3cm}|}
\hline
記号&意味\\\hline\hline
\texttt{l}& 左寄せ\\\hline
\texttt{|}& 縦の罫線を入れる　　\\\hline
\texttt{p}& 指定した幅の段落を入れる\\\hline
\texttt{*}&　　フォーマットの繰り返し\\\hline
\end{tabular}
```

記号	意味
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

上記の例では 2 行にわたって表示されている部分では上下の線に内容が近づきすぎていると感じる人もいるかもしれません。上下の行間隔を変えるためにはその分だけ見かけの行の高さ (厳密に言えばベースラインから上の部分の長さである高さとその下の部分の長さである深さ) を変更します。このためには幅が 0 である罫線を入れるというのが普通です。

```
\begin{tabular}[t]
{|c|>{\rule{0ex}{3ex}}m{3cm}
      <{\rule[-1.5ex]{0ex}{3ex}}|}
\hline
記号&意味\\\hline\hline
\texttt{l}& 左寄せ\\\hline
\texttt{|}& 縦の罫線を入れる     \\\hline
\texttt{p}& 指定した幅の段落を入れる\\\hline
\texttt{*}&      フォーマットの繰り返し\\\hline
\end{tabular}
```

記号	意味
l	左寄せ
|	縦の罫線を入れる
p	指定した幅の段落を入れる
*	フォーマットの繰り返し

ここでは 2 番目の項目の前に>を用いて幅が 0ex で高さが 3ex の、後に<を用いて幅が 0ex で高さが 3ex のものを-1.5ex 上げて (1.5ex 下げて) 配置しています。したがって、解説文の一番上では上に空く空間が、最後では下に空く空間がそれぞれ確保されることになります。

\rule は次の形をとります。

```
\rule[上への移動距離]{幅}{高さ}
```

行の間隔が狭くなったときに調整する一般的な方法です。

4.3 表組みの Tips

表組みに関する細かい注意は [8] の解説を見てください。この文献には他書にない、表組みのデリケートな部分の解説があります。

ここでは表組みに関するいくつかの基礎について解説します。

4.3.1 部分的に横線を入れる

部分的に横線を入れるためには \hline の代わりに \cline を用います。次の例を見て下さい。2 行目のところに部分的な横線が入っています。

```
\begin{tabular}{|c|c|c|c|c|}
 \hline
初めの項&2 番目の項 &3 番目の項 &4 番目の項 & 5 番目の項
 \\\cline{2-2}\cline{4-5}
 注意 1& 次の話題& 機能の解説& 戻る& \\ \hline
\end{tabular}
```

初めの項	2 番目の項	3 番目の項	4 番目の項	5 番目の項
注意 1	次の話題	機能の解説	戻る	

4.3.2　行方向にいくつかの項をまとめる

行方向にいくつかの項をまとめるためには \multicolumn を用います。なお、対応する項目の次の項目の直前までの部分がこの \multicolumn で設定した内容に置き換えられています。

```
\begin{tabular}{|c|c|c|c|c|}
 \hline
初めの項&2 番目の項 &3 番目の項 &\multicolumn{2}{c|}{4 番目と 5 番目の項}
 \\\cline{2-2}\cline{4-5}
 注意 1& 次の話題& 機能の解説& 戻る& 注文\\ \hline
\end{tabular}
```

初めの項	2 番目の項	3 番目の項	4 番目と 5 番目の項	
注意 1	次の話題	機能の解説	戻る	注文

次の例では単価のところに毎回「円」を書くために@{ 円 } を入れています。しかし、項目名のところにも「円」が入ってしまいます。

```
\begin{tabular}[t]{|c|r@{円}|}
\hline
品物&単価\\\hline\hline
CPU & 20000\\ \hline
メモリー&5000 \\ \hline
 ケース& 8000\\ \hline
 DVD ドライブ& 9200\\ \hline
 マウス& 2000\\ \hline
 キーボード& 1100\\ \hline
\end{tabular}
```

品物	単価円
CPU	20000 円
メモリー	5000 円
ケース	8000 円
DVD ドライブ	9200 円
マウス	2000 円
キーボード	1100 円

表の項目名のところの表示を変えるために 1 コラムの \multicolumn を用いることができます。表題の部分を中央ぞろえに変更するために利用しています。

```
\begin{tabular}[t]{|c|r@{円}|}
\hline
品物&\multicolumn{1}{c|}{単価}\\
\hline\hline
CPU & 20000\\ \hline
メモリー&5000 \\ \hline
 ケース& 8000\\ \hline
 DVD ドライブ& 9200\\ \hline
 マウス& 2000\\ \hline
 キーボード& 1100\\ \hline
\end{tabular}
```

品物	単価
CPU	20000 円
メモリー	5000 円
ケース	8000 円
DVD ドライブ	9200 円
マウス	2000 円
キーボード	1100 円

4.3.3 列方向にいくつかまとめる

1 つのセルを縦方向にいくつかまとめるのには multirow パッケージで定義されている \multirow 環境を利用するのが一番楽です (\usepackage{multirow} を指定します。)。

```
\begin{tabular}[t]{|c|c|c|c|c|}
 \hline \multirow{2}{2zw}{氏名}
&\multicolumn{2}{c|}{5 月 10 日} &
 \multicolumn{2}{c|}{5 月 11 日}\\\cline{2-5}
&午前 &午後 &午前 &午後 \\\hline
 平野&$\bigcirc$& $\times$ &$\times$  & $\triangle$\\ \hline
\end{tabular}
```

氏名	5 月 10 日		5 月 11 日	
	午前	午後	午前	午後
平野	◯	×	×	△

- \multirow は列方向の数、その幅の 2 つを引数の取ります。
- 2 行にわたって表示したいので初めの引数に 2 を指定しています。
- 表示する「氏名」が漢字 2 文字なので幅として 2zw(zw は全角文字の幅) を指定しています。

なお、この例では \bigcirc などの記号を表記するために数式モードを利用しています。このモードの使い方については第 6 章を見てください。

4.3.4 表の罫線の幅を変える

上記の例では表の罫線はすべての場所で同一です。部分的に変えるには次の例を見てください。

```
\begin{tabular}[t]{%
@{\vrule width 1pt}c!{\vrule width 1pt}r%
!{円\vrule width 1pt}}
\noalign{\hrule height 1pt}
品物&\multicolumn{1}{c@{\vrule width 1pt}}{単価}\\
\noalign{\hrule height 1pt}
CPU & 20000\\ \hline
メモリー&5000 \\ \hline
 ケース& 8000\\ \hline
 DVD ドライブ& 9200\\ \hline
 マウス& 2000\\ \hline
 キーボード& 1100\\
\noalign{\hrule height 1pt}
\end{tabular}
```

品物	単価
CPU	20000 円
メモリー	5000 円
ケース	8000 円
DVD ドライブ	9200 円
マウス	2000 円
キーボード	1100 円

- 横線の幅を変えるために \\ の直後に次のマクロを入れています。

  ```
  \noalign{\hrule height 1pt}
  ```

これにより 1pt(1/72.27 インチ=0.351mm) の幅の水平線が入ります。

- 垂直の線の幅を変えるために tabular 環境の列の形式の中に@で次のものを挿入しています。

  ```
  \vrule width 1pt
  ```

  array 環境で定義されている!を挿入するとデフォルトで入る列間の空白が取り除かれるません。

- これらの線の変更は複数の場所に現れるので体裁の統一や、読みやすさを考慮すると後で述べるマクロを用いたほうがよいでしょう。たとえば

  ```
  \newcommand{\WVrule}{\vrule width 1pt}
  ```

  を table 環境の前に書いておけば上記のコマンドの代わりに \WVrule と書くだけですみます。罫線の幅を変えたくなったらここの 1pt の値を変えるだけですみます。

  マクロについては第 7 章を参照してください。

やってみよう 4.1 上記の例で*!*を@に変えたらどのように変化するか調べなさい。

やってみよう 4.2 九九の表を見やすく書き直しなさい。

4.4 table 環境

一般に、表には表題や通し番号がつきます。tabular 環境では表の構成をするだけなので表題などをつけるためには table 環境の中に tabular 環境を書くことになります。表題は \caption コマンドを用います。先ほどの表に表題をつけて見ましょう。

```
\begin{table}[ht]
\caption{平野研究室セミナースケジュール調整表}
\begin{center}
\begin{tabular}[t]{|c|c|c|c|c|}
 \hline \multirow{2}{2zw}{氏名}
&\multicolumn{2}{c|}{5 月 10 日} &
 \multicolumn{2}{c|}{5 月 11 日}\\\cline{2-5}
&午前 &午後 &午前 &午後 \\\hline
 平野&$\bigcirc$& $\times$ &$\times$  & $\triangle$\\ \hline
\end{tabular}
\end{center}
\end{table}
```

この例では表を中央に置くために center 環境の中に入れています。また、tabular 環境の前に表題を記述するようにしています。

表 4.1: 平野研究室セミナースケジュール調整表

氏名	5 月 10 日		5 月 11 日	
	午前	午後	午前	午後
平野	○	×	×	△

4.5 ページをまたぐ表を作成する

1 ページに入りきらないような大きな表を作成する場合には supertab や longtable などのパッケージで定義されている環境を使うことができます。

longtable では longtable 環境が定義されています。この環境は次のような特徴があります。

- 異なるページであっても対応する列の幅がすべて同じになるように設定されます。幅が同じになっていない間は警告がでます。
- 各ページの先頭や末尾の共通の行は 1 回だけ指定すれば十分です。
- 各行の形式を定義する方法は tabular と同じです。
- \hline や \cline も使えます。

longtable を利用するため文書のプリアンブルで \usepaakage を用いて利用を宣言してください。

longtable は次のような形をとります。

```
\usepackage{longtable}
\begin{longtable}{cccp{15em}}
    \caption{longtable の例}\\
    拡張子&作成アプリケーション&実行アプリケーション名&
      \multicolumn{1}{c}{用途}\\ \hline
  \endfirsthead
    \caption{longtable の例 (続き)}\\
    拡張子&作成アプリケーション&実行アプリケーション名&
      \multicolumn{1}{c}{用途}\\ \hline
  \endhead
    \hline \multicolumn{4}{r}{次ページへ続く}
  \endfoot
    \hline
  \endlastfoot
    \texttt{.txt} & メモ帳 (\texttt{notepad.exe})& 特になし&テキストファイル作成\\
\end{longtable}
```

- 先頭で tabular 環境と同じように列のフォーマットを指定します。

- はじめに表の先頭に入れるものを宣言します。ここでは \caption と第 1 行目の各列の題目を指定しています。
- 最後に \endfirsthead を入れて先頭の表題が終了することを宣言します。
- 次に 2 ページ目以降の第 1 行目の表題を記述します。ここでは一番初めのものと同じ (\caption 部分は字句が追加されています) を入れます。
- 最後に \endhead を入れます。
- 表の各ページの一番の下に記述する項目を書きます。ここでは横の罫線と「次のページへ続く」を入れました。
- 最後に \endfoot を入れます。
- 表の一番最後に記述する項目を \endlastfoot までに入れます。ここでは横の罫線を入れています。
- 途中の項目は tabular 環境の場合と同じように&で区切り、行の最後に \\ を入れます。

表 4.2: longtable の例

拡張子	作成アプリケーション	実行アプリケーション名	用途
.txt	メモ帳 (notepad.exe)	特になし	テキストファイル作成
.tex	テキストエディタ	platex など	LaTeX TeX のファイル
.dvi	platex	プレビューア	platex などで処理された後の整形された文書形式
.log	platex	特になし	実行時の詳しい情報が入る
.toc	platex	特になし	目次の情報が入る
.lof	platex	特になし	図目次の情報が入る
.aux	platex	特になし	ラベルなどの情報が入る
.idx	platex	makeindex など	索引の情報が入る
.ind	makeindex など	platex	フォーマット化された索引
.ps	テキストエディタで可	ghostview など	ページ記述言語 PostScript で書かれた、主に図形を記述するファイル
.eps	Illustlator など		画像の大きさの情報が入っている PS ファイル。ghostview は.ps ファイルを.eps ファイルに変換する機能を持つ
.c	テキストエディタで可		C 言語で書かれたファイル
.h	テキストエディタで可		C 言語で各種定義が書かれたファイル。.c に読み込まれて使用される
.exe	各種コンパイラ	自分自身	Windows の実行可能ファイル

次ページへ続く

表 4.2: longtable の例 (続き)

拡張子	作成アプリケーション	実行アプリケーション名	用途
.bat	テキストエディタで可	自分自身	いくつかのコマンドを順次実行するように支持するためのファイル
.pl	テキストエディタで可	perl	簡単なデータ処理を目指したインタープリタ言語。Web で CGI 言語として利用することで広まった。拡張子は特に指定はない
.php	テキストエディタで可	apatche 内のモジュール、または php	サーバーサイドで動く HTML 文書内に書くことが可能な言語。CGI 言語として利用される
.xml	テキストエディタで可	ブラウザーほか	タグを自由に定義できる情報を記述する。最近ではインターネット上でのアプリケーション間のデータ交換のフォーマットとして標準になっている。
.html	テキストエディタで可	ブラウザー	Hyper Text Markup Language の略。ホームページを記述するための言語として利用される。
.xhtml	テキストエディタで可	ブラウザー	厳密な XML の文法にしたがった HTML 文書
.js	テキストエディタで可	ブラウザー	HTML 文書をクライアントサイドで変化させるために利用される言語 JavaScript のファイル
.css	テキストエディタで可	ブラウザー	HTML 文書の表示を記述するための Cascading StyleSheet を記述したファイル

この例から次のことがわかります。

- 表はその場に書かれています。
- 改ページされた表の先頭に別の表題がつけられていますが番号は当然同じです。

supertab には文献 [8] で調べてください。ここには両者の比較も載っています。

やってみよう 4.3 複数ページにわたる表を作成してみましょう。

第5章　図の取り扱い

LaTeX には簡易な 2D の図を書く機能があります。単純な図はこれでもかまいませんが込み入った図や、他のソフトウェアで作成した図を取り込む機能もあるので必要に応じて使い分ける必要があります。

5.1　picture 環境

5.1.1　picture 環境の基礎

picture 環境は次の形をとります。

```
\begin{picture}(図形全体の横幅, 図形全体の高さ)(左下の x 座標, 左下の y 座標)
 <配置するもの>
\end{picture}
```

ここで図形の大きさや座標の位置の単位は \unitlength ディメンションレジスターに指定します。\setlength{\unitlength}{1cm} のように指定します。この中 \put を用いて位置を指定しておくものを書きます。

```
\put(x 位置,y 位置){置くもの}
```

位置の単位は \unitlength で与えられているので位置指定の値には単位を付けません。書き出すためのものは次のように指定します。

- 文字列
  単に文字列を書いてもかまいませんが、指定した位置が文字列の右端や中央にするのであれば \makebox を使うのがよいでしょう。

  ```
  \makebox[幅][位置]{文字列}
  ```

  の形をとります。幅と位置の指定はオプションです。出力される文字列より幅が狭くてもかまいません。

- 直線
  直線を引くのには \line を用います。注意してほしいのは任意の傾きの直線を描くことはできないということです。

  ```
  \line(x 方向,y 方向){長さ}
  ```

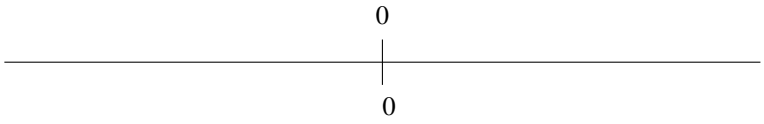
の形です。x 方向、y 方向は直線の向きを表します。したがって、(2,2) と (1,1) では同じ方向になります。これらの値は $-6,-5,\ldots,5,6$ の値しかとることができません。

- 円
  円を描くのには \circle を用います。\circle*とすると中が塗りつぶされます。

  \circle{半径}

  の形をとります。

```
{\setlength{\unitlength}{1cm}
\begin{center}
  \begin{picture}(10,3)(-5,-1.5)
    \put(-5,0){\line(1,0){10}}
    \put(0,-0.3){\line(0,1){0.6}}
    \put(0,-0.7){$0$}
    \put(0,0.5){\makebox[0em]{$0$}}
  \end{picture}
\end{center}
}
```



- 長さの単位を 1cm にしています (1 行目)。この始まりに { がついているのは次の picture だけにこの長さの単位を使いたいからです。例の最後に } があることに注意してください。
- 図の大きさは横 10cm、縦 3cm としていて右下の座標位置を $(-5,-1.5)$ としています (3 行目)。
- 長さが 10 の水平な直線を $(-5,0)$ の位置から書いています (4 行目)。
- 直線の中央に長さが 0.6 の垂直な直線を引いています。
- この位置を数直線の原点としたいので 0 を表示しています。ここでは \makebox の効用を見るために 2 つ書いています。
- 上に書いてあるほうに \makebox を用いています。このボックスは長さを 0 にしているので文字の中央と垂直な直線の位置が縦方向にそろいます。

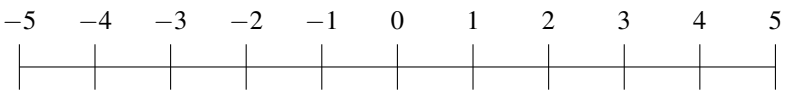
5.1.2 繰り返しの出力

上記の例でさらに目盛りと数字を書き込もうとすると書き込む分だけ \put を必要とします。同じパターンの繰り返しは \multiput を用いると簡単に書けます。

\muliput(x 位置,y 位置)(x 位置の移動量,y 位置の移動量){繰り返しの回数}{置くもの}

です。これを用いると次のような目盛りを付けることができます。

```
{\setlength{\unitlength}{1cm}
\newcounter{Tmp}
\setcounter{Tmp}{-5}
\begin{center}
  \begin{picture}(10,3)(-5,-1.5)
    \put(-5,0){\line(1,0){10}}
    \multiput(-5,-0.3)(1,0){11}{\line(0,1){0.6}}
    \multiput(-5,0.5)(1,0){11}{%
        \makebox[0em]{$\arabic{Tmp}$}\addtocounter{Tmp}{1}}
  \end{picture}
\end{center}
}
```



- 前の例で

  \put(0,-0.3){\line(0,1){0.6}}

  となっていたところが

  \multiput(-5,-0.3)(1,0){11}{\line(0,1){0.6}}

  に変わっています。(−5,0.3) から (1,0) づつ変化して 11 個の垂直な直線がかかれることになります。

- −5 から 5 までの数字を出力するためにも \multiput を用いています。1 づつ増加する値を出力するためにカウンターを用いています。

- 2 行目で新しいカウンター Tmp を定義しています。その初期値として −5 を代入します (3 行目)。

- カウンターの値を表示した後(\arabic{Tmp})、その値を 1 増やしています(\addtocounter{Tmp}{1})

この例でカウンターの値を出力する \multiput の行が長いので途中で改行しています。改行の途中でコメントを示す%が入っていますがこれは出力に無駄な空白が入らないようにするためです。

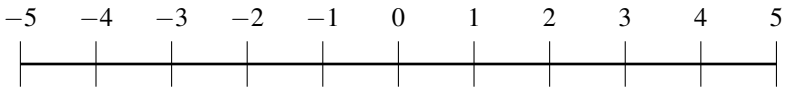
やってみよう 5.1 %を取り除いた場合の出力が変化することを確認しなさい。

やってみよう 5.2 値が負の場合には − を取り除いた部分の数字が垂直な直線上にくるように直しなさい。

やってみよう 5.3 幅が *1mm* の縦横 *5cm* の方眼紙を書きなさい。方眼紙の目盛りは *1cm* 毎に太くすること。

なお、この例では直線の幅がすべて同じです。直線の幅を変化させるには \linethickness を用いるとよいでしょう。

```
{\setlength{\unitlength}{1cm}
\newcounter{Tmpp}
\setcounter{Tmpp}{-5}
\begin{center}
  \begin{picture}(10,3)(-5,-1.5)
    \linethickness{1pt}
    \put(-5,0){\line(1,0){10}}
    \thinlines
    \multiput(-5,-0.3)(1,0){11}{\line(0,1){0.6}}
    \multiput(-5,0.5)(1,0){11}{%
        \makebox[0em]{$\arabic{Tmpp}$}\addtocounter{Tmpp}{1}}
  \end{picture}
\end{center}
}
```



ここでは横の直線の幅を 1pt にしています。また、元の幅に戻すために \thinlines を用いています。

5.2 graphicx パッケージ

LaTeX には他のソフトウェアで作成された図を取り込み機能があります。この機能を使うためには graphicx パッケージが必要です。したがって、文書のはじめ (プリアンブル) に

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

と書いておくことが必要です。このオプションのパラメータ [dvips] は dvi ファイルを PostScript に変換するためのプログラムの名前です。ここでは dvips を用いることになります。graphicx を用いると \includegraphics で画像ファイルを取り込むことが可能です。

ここでは GhostScript の配布に含まれている escher.ps を表示してみましょう。このファイル形式が ps なので GhostView で eps に変換します。このファイルを GhostView で表示しておき、

$$\text{File} \Longrightarrow \text{PS to EPS}$$

を選択します。いくつか質問されますがすべて「はい」でかまいません。ファイル名をここでは escher.eps として TEX の文書と同じフォルダに保存しましょう。文書には次のように書きます。

```
\includegraphics[width=10cm]{escher.eps}
```

ここでは表示する幅 (width) を 10cm に指定しています。この場合、高さは eps ファイルの中に書

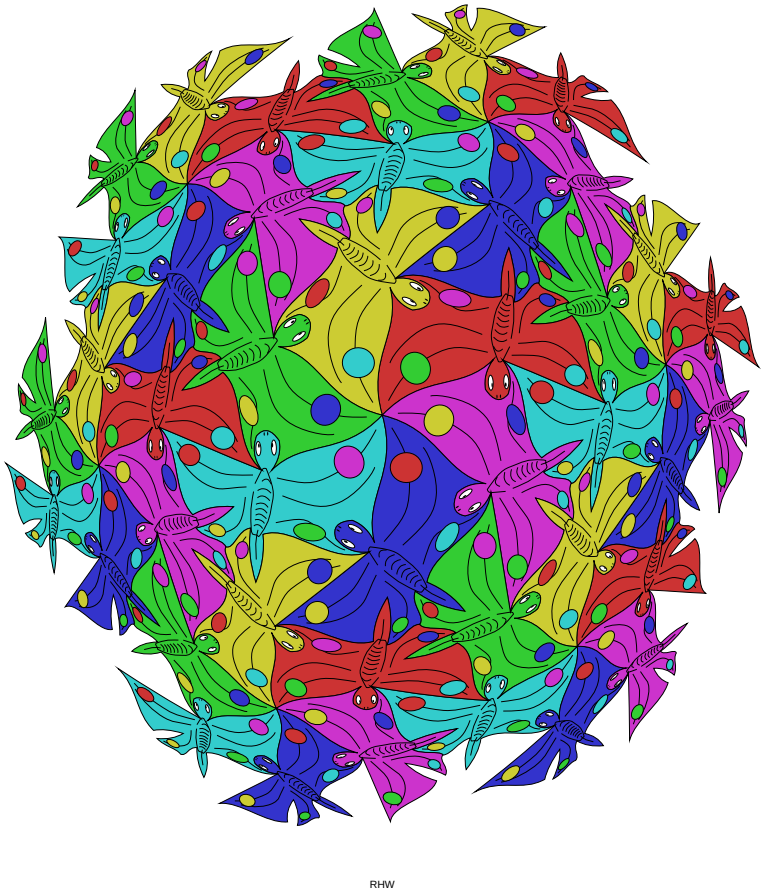


図 5.1: GhostScript の配布に含まれている PS ファイルの表示

かれている画像の大きさの情報にしたがって調整されます。同様に height により高さを指定できます。両方を指定した場合には指定した値に設定されます。この場合、一般には画像が歪みます。

やってみよう 5.4 いろいろな画像を取り込んで大きさを変えて表示してみましょう。

直接の幅などを指定する代わりに与えられた画像を拡大したり縮小したりする \scalebox があります。

\scalebox{拡大率}[横方向の拡大率]{画像ファイル名}

オプションの横方向のパラメータがない場合には縦と横の拡大率は同じ値に設定されます。
　また、このパッケージでは \rotatebox も用意されています。

\rotatebox{回転角度}{表示するもの}

の形をとります。オプションの引数をとることもできますがここでは省略します。回転角の単位は度です。回転方向は反時計回りです。
　画像の中に説明の文字を入れたりするのには picture 環境の中で \includegraphics を用いて画像を取り込み、\put で文字列を表示するという方法がお勧めです。この方法で図を取り込んだものが次の例です。

```
\newcommand{\PutStringwithArrowY}[2]{%
\rotatebox{90}{\makebox[#1\unitlength][l]%
     {$\longleftarrow$\hfill%
       \rotatebox{-90}{\makebox[0em][c]{#2}}\hfill$\longrightarrow$}}}
\newcommand{\PutStringwithArrowX}[2]{%
  \makebox[#1\unitlength][l]{%
    $\leftarrow$\hfill{#2}\hfill$\rightarrow$}}
\begin{figure}[h]
\begin{center}
{\setlength{\unitlength}{1.5cm}
\begin{picture}(5,8)(-1.5,-0.5)
  \put(-1.55,-0.55){\scalebox{1.5}{\includegraphics{ps/graphx2+1.eps}}}
  \put(3.6,-0.05){$x$}
  \put(0,6.6){$y$}
  \put(2.3,6.6){$C$}
  \put(3.3,6.6){$L$}
  \put(1.0,1.7){\PutStringwithArrowX{1}{$dx$}}
  \put(1.0,0.2){\PutStringwithArrowX{1}{$\Delta x$}}
  \put(1.0,-0.3){\makebox[0em][c]{$x$}}
  \put(2.0,-0.3){\makebox[0em][c]{$x+\Delta x$}}
  \put(1,2){\line(1,0){2}}
  \put(2,4){\line(1,0){0.5}}
  \put(2,5){\line(1,0){1}}
  \put(1,0){\line(0,1){2}}
  \put(2,0){\line(0,1){5}}
```

```
  \put(2.2,2){\PutStringwithArrowY{2}{$dy$}}
  \put(2.7,2){\PutStringwithArrowY{3}{$\Delta y$}}
 \end{picture}%
}
\end{center}
\caption{図にテキストをつけ加えた例}\label{graphx2+1}
\end{figure}
```

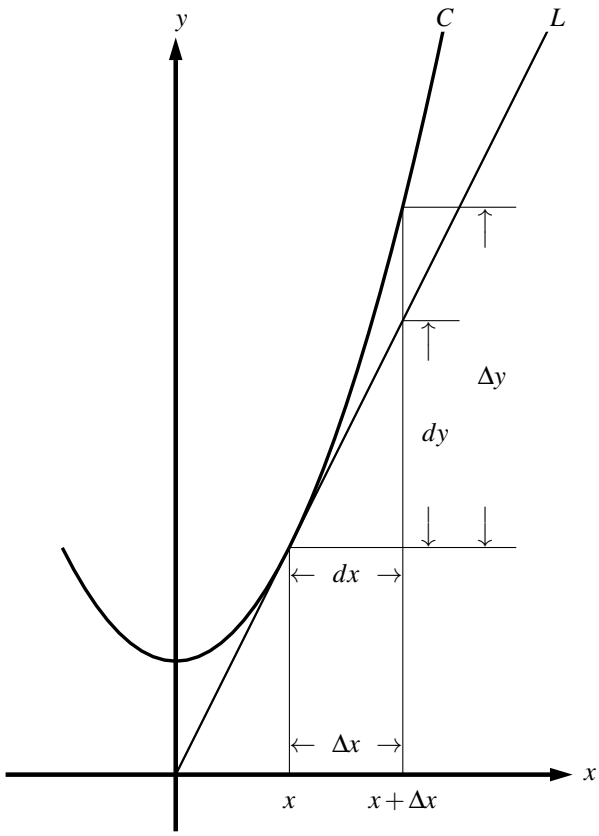


図 5.2: 図にテキストをつけ加えた例

この例では放物線と座標軸を PostScript で記述し、残りの補助線と説明のための文字列を \put で記述しています。なお、この例は次の節で解説する figure 環境に含まれていることに注意してください。

- picture 環境における長さの単位は 1.5cm にしています。作成した PS ファイルでは長さの単位を 1cm にしてあるので \includegraphics で取り込まれる画像を \scalebox を用いて 1.5 倍に拡大しています。

- 両側に矢印がついた文字列を 2 回書くのでマクロを定義しています (\PutStringwithArrow など)。このマクロは入れるためのボックスの幅と中に入れる文字列を引数にしています。文字列の両端に \hfill を入れることで矢印を端がつくようにしています。

なお、取り込んだ PS ファイルのリストは付 録 Aにあります。

5.3 figure 環境

figure 環境は atable 環境と同様に図に対して見出し (\caption) を付ける機能を持っています。図 5.2では図を中央に置くために center で囲み、それを figure 環境で囲んでいます。

なお、LaTeX には図目次を表示するためのマクロ \tableoffigure があります。

5.4 PostScript ファイルへの変換

画像を LaTeX の文書に取り込むときそれが表示できるかどうかはプレビューアに任されています。最終的な配布状況や、プレビューアの設定を考えると PostScript ファイルへ変換しておくのが一番簡単なように思われます。各種コンバータは TeXWiki[1] の「変換ツール」を参照してください。

5.4.1 jpeg ファイルを PostScript ファイルに変換する

jpeg2ps.exe による変換　変換の方法は

```
jpeg2ps 変換したいファイル.jpg > 変換後のファイル.eps
```

です。> はリダイレクションと呼ばれる記法で画面に出力[2] されるものを指定されたファイル内にしまうという意味です。

Photoshop を利用する　Adobe 社の Photoshop は eps ファイルでの出力が可能です。したがって、Photoshop で開ける画像はすべて eps ファイルに変換できることになります。同様のソフトウェアはフリーでもあります [1]。

jpeg 画像を直接取り扱う　jpeg 画像などを直接取り扱うことも可能です。このときは、dvioutと pdf ファイルに変換した後では扱いが異なります。

pdf ファイルに jpeg ファイルを埋め込むときは画像の大きさを指定する必要があります。図 5.3は図 5.1 を jpeg ファイルの変更したものです。

この部分の表示は次のようになっています。

```
\scalebox{0.5}{\includegraphics[bb= 0 0 577 674]{escher.jpg} }
```

[1] http://oku.edu.mie-u.ac.jp/ okumura/texwiki/
[2] 正確には標準出力です。

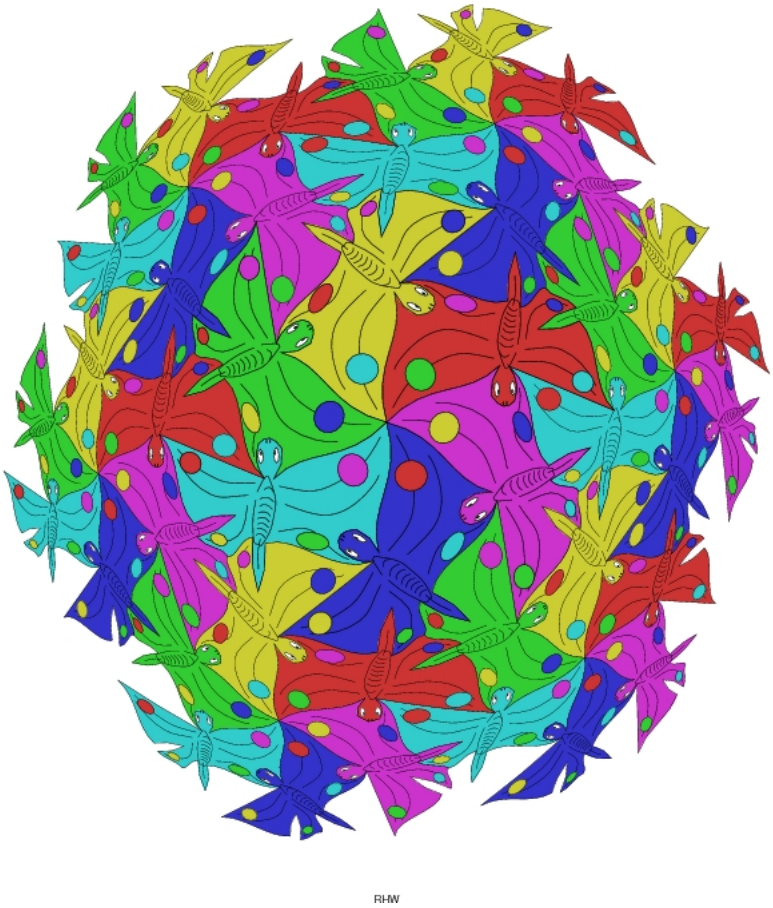


図 5.3: GhostScript の配布に含まれている PS ファイルの表示 (jpg 版)

`\includegraphics` のパラメータで `bb` は BoundingBox の略です。画像の左下と右上の座標位置を画素単位 (pixel) で指定します。この場合は右下の位置は `0 0` でよく、右上の座標は画像ファイルの大きさを指定することになります。この大きさはエクスプローラ上でファイルの上にカーソルを載せる確認することができます (図 5.4)。なお、`\scalebox` で大きさを 0.5 倍に縮小しているのはこの画像を 72dpi(dot per inch) で表示したときの大きさが

$$577/72 * 2.54 = 20.355\ldots 1\text{inch} = 2.54\text{cm}$$

となるので前の図の横幅 (10cm) と合わせるためです。

この形式では `dviout` で図を表示できません。

5.4.2 コンピュータの画面のキャプチャー

この文書でははじめのほうにコンピュータの画面を取り込んでいます。この方法は次のとおりです。

図 5.4: jpg ファイルの画像の大きさの確認

- 取り込みたいアプリケーションの画面をアクティブにします。
- `Alt` キーと `Print Screen` キーを同時に押します。これによりアクティブウィンドウがクリップボードにコピーされます。
- 画像を貼り付けるツールを起動します。筆者は Photoshop を用いています。Windows に付属の「ペイント」でも十分です。
- 「ファイル」⇒「新規」を選び、新しいウィンドウを開きます。
- 「編集」⇒「貼り付け」で先ほどのアクティブウィンドウの画像が表示されます。
- 「ファイル」⇒「名前を付けて保存」で保存形式を `jpeg` などにして適当な名前を付けて保存します。

この方法でアプリケーションがメニューを開いている状態をクリップボードにコピーすることは残念ながらできません。これは `Alt` キーを押したときにメニューが閉じられてしまうからです。`Print Screen` キーだけ押すと全画面がクリップボードにコピーされるのでそこから必要な部分を切り出してください。

このほかにもアクティブウィンドウの画像を直接 EPS ファイルとして保存するフリーなソフトウェアも存在します。ここでは WinShot(`http://www.woodybells.com/winshot.html`) を紹介しておきます。

第6章　数式の書き方

数式をきれいに出力するという機能は TEX の特徴のひとつです。数式モードでは空白はマクロの間の区切りという意味しか持ちません。数式は独自のスペースで組み上げます。特別のことがない限り、TEX のデフォルトの形を用いましょう。

6.1　数式モードの違い

TEX(LATEX) は数式に対しては数式モードにおいて記述することになります。数式モードでは通常の文章と異なるフォントを使用し、文字の配置も別の方式をとります。また、数式モードには本文の中に記述する場合と、別の行に数式を記述するモードの二つがあります。文中モードの場合には表示したい数式を二つの`$`の間に書きます。独立の式を書くモードにはいろいろ場合があります。文中に書く場合には縦方向にあまり幅をとらないように配置されます。例 6.1 を見てください。

表 6.1: 文中モードと別立てモードの違い

入力	文中モード	別立てモード
`\frac{1}{x+2}`	$\frac{1}{x+2}$	$\dfrac{1}{x+2}$
`\int_a^b f(x)dx`	$\int_a^b f(x)dx$	$\displaystyle\int_a^b f(x)dx$
`\sum_{k=1}^{n}k`	$\sum_{k=1}^{n} k$	$\displaystyle\sum_{k=1}^{n} k$
`\lim_{n\rightarrow\infty}(\sqrt{n^2+1}-n)`	$\lim_{n\rightarrow\infty}(\sqrt{n^2+1}-n)$	$\displaystyle\lim_{n\rightarrow\infty}(\sqrt{n^2+1}-n)$

なお、分数を表す `\frac` には文章内でも別立て行の形式で表示する `\dfrac` というマクロもあります。また、同様のことは `\displaystyle` を用いることでも可能です。

6.2　簡単な数式の書き方

6.1 でも簡単な数式の書き方を示しましたがより詳しく説明をします。

6.2.1　上付きと添え字

べき乗 (上付き) の書き方　上付きにする場合には^を用います。この後に来る文字は { と } で囲むのが一般の使い方です。なお、は^を続けて書くとエラーになります。たとえば e^{x^2} と書くには `e^{x^2}` と書きましょう。

添え字の書き方　`a_{1}` における _ は次のグループの部分を下付き (添え字) にすることを指示します。

上付きと添え字がある場合　上付きと添え字を同時に指定する場合にはどちらを先にしても同じ結果が得られます。

上付きと添え字はどちらを先に書いても同じです。
```
\[
 x_1^2 = x^2_1 = {x_1}^2
\]
```
添え字の後に上付きを置きたければ添え字までの部分をグループ化します。

上付きと添え字はどちらを先に書いても同じです。

$$x_1^2 = x_1^2 = {x_1}^2$$

添え字の後に上付きを置きたければ添え字までの部分をグループ化します。

組み合わせの数の書き方　n 個のものから r 個選ぶ組み合わせの数 ${}_nC_r$ のように後に添え字などの本体が来る場合には `{}_nC_r` のように書きます。

6.2.2　分数の書き方

分数を書くためには `\frac` を用います。`\frac{分子}{分母}`と書きます。分子や分母の文字列の長さが 1 のときは`{}`ではさむ必要はないのですが習慣として常につけるようにしておくことをお勧めします。

分子や分母の中に分数を書いても
フォントの大きさは自動的に調整されます。
```
\[ \frac{1}{1+\frac{1-t^2}{1+t^2}} \]
```
これでは分母の式が見にくいと思うのであれば
`\verb+\dfrac+`を用います。
```
\[  \frac{1}{1+\dfrac{1-t^2}{1+t^2}} \]
```

分子や分母の中に分数を書いてもフォントの大きさは自動的に調整されます。

$$\frac{1}{1+\frac{1-t^2}{1+t^2}}$$

これでは分母の式が見にくいと思うのであれば`\dfrac` を用います。

$$\frac{1}{1+\dfrac{1-t^2}{1+t^2}}$$

6.2.3 総和などの書き方

6.1 にあるように TEX では数学的な記号についてはマクロが定義されているのでそれを用います。総和における下限や上限の値、定積分の上端や下端の指定はそれぞれ添え字や上付きの指定と同じように書きます。TEX が適切な位置にそれらの文字列を置いてくれます。対数の底も添え字で実現します。

数式の中で関数名はイタリックにはなりません。このことから関数名についてもマクロが定義されています。

表 6.2: 数学の記関数号

`\sin`	`\cos`	`\tan`	`\cot`	
`\arcsin`	`\arctan`	`\sec`	`\log`	`\ln`
`\sinh`	`\cosh`	`\tanh`	`\lim`	

導関数を表す $'$ は ’ を用います。$(\tan x)' = \dfrac{1}{\cos^2 x}$ は `(\tan x)'=\dfrac{1}{\cos^2x}` と書きます。また、根号には `\sqrt` を用います。この後にオプションをつけると根号の前に小さく着くので 3 乗根などの記述ができます。

やってみよう 6.1 次の数式を書いてみましょう。

1. $\dfrac{x+y}{2}$
2. $\dfrac{d}{dx}\sin x = \cos x$
3. $(x+1)^2 = x^2+2x+1$
4. $\sqrt[3]{27} = 3$

6.3 別立ての数式の書き方

別立ての式を書くモードの中では空行を入れるとエラーになる場合が多いので見やすくするためには行の先頭に % を入れましょう。

別立ての数式を 1 行だけ書く 別立ての数式を 1 行だけ書くためには `\[`と`\]` の間に数式を書きます。数式は中央ぞろえになります。

```
\[1+2+3+\cdots+n=\frac{n(n+1)}{2}\]
```

$$1+2+3+\cdots+n=\frac{n(n+1)}{2}$$

ここで使われている `\cdots` は中央に 3 つの点を打つためのマクロです。

式に参照の番号をつける　式に番号をつけるためには equation 環境か、align 環境を用います。[1] これらの環境で数式に番号をつけたくない場合には \nonumber を行のところに入れます。

```
\begin{align}
 x'&=x\cos \theta+y\sin\theta\nonumber\\
 y'&=-x\sin\theta+y\cos\theta\label{formula}
\end{align}
```

$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta \tag{6.1}$$

ここではこの式の番号を別のところで参照できるように \label でラベルをつけています。これを \ref{formula} とすることでこの番号が参照できます。参照できるのは式の番号だけなので括弧などは自分でつける必要があります。

また、ページで参照したい場合には \pageref{formula} を用います。

はじめの式には \nonumber がついているので式に番号がつきません。

複数行にわたる式で参照の番号をつけない　すべての行に入れたくない場合には align*環境を用います。

```
\begin{align*}
 x'&=x\cos \theta+y\sin\theta\nonumber\\
 y'&=-x\sin\theta+y\cos\theta
\end{align*}
```

$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta$$

ここでは \nonumber を使わなくても式の番号がつきません。

6.4　いろいろな記号の書き方

数式の中で使われるギリシャ文字は次のように書きます ($ ではさむか数式モードでしか使えません)。この表にないギリシャ文字の大文字はアルファベットで代用します。

不等号や演算子の記号は表 6.4 を参考にしてください。定義されている記号はこれですべてではありません。必要があれば別の図書で調べてください。なお、ここにある記号のうちいくつかは amsmath や amssymb パッケージを読み込まないと使用できません。

6.5　内容に応じたおおきな括弧を作る

TEX では括弧の大きさを自動で調整してくれる機能があります。そのためのマクロが \left(任意の括弧) と\right(任意の括弧) です。同じ種類の括弧をつける必要がありません。どちらかの括弧が必要ないときは . を用います。同じグループ内で\left と\right が対応していないといけません。長い数式を途中で改行するときなどは気をつけましょう。

[1] 以前の LATEX では複数の数式を一定の位置でそろえて表示する環境として eqnarray がありました。この環境ではそろえる位置の前後に追加される空白が少し多く、数式の見栄えがよくありません。最近では align 環境を使うのが常識のようです。

表 6.3: ギリシャ文字

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

表 6.4: 演算子と比較演算子

$\pm$	\pm	$\mp$	\mp	$\times$	\times	$\div$	\div	$\equiv$	\equiv
$\bullet$	\bullet	$\cdot$	\cdot	$\circ$	\circ				
$\cup$	\cup	$\cap$	\cap	$\in$	\in	$\ni$	\ni		
$\subset$	\subset	$\supset$	\supset	$\subseteq$	\subseteq	$\supseteq$	\supseteq		
$=$	=	$\leq$	\leq	$\geq$	\geq	$\neq$	\neq	$\sim$	\sim
$\leqq$	\leqq	$\geqq$	\geqq						

```
\begin{eqnarray*}
 \left(\frac{x^2}{x+1}\right)^3\\
f(x)&=&\left\{
\begin{array}{l@{\hspace{2em}}l}
    x\sin \dfrac{1}{x} & x\neq0 のとき\\
    0&x=0 のとき \\
\end{array}
\right.
\end{eqnarray*}
```

$$\left(\frac{x^2}{x+1}\right)^3$$

$$f(x) \quad = \quad \left\{ \begin{array}{ll} x\sin \dfrac{1}{x} & x\neq0 \text{ のとき} \\ 0 & x=0 \text{ のとき} \end{array} \right.$$

この例では関数の定義のところで array 環境を用いています。この環境は tabular 環境と同じ使い方をしますが、数式モードの中でしか使えません。その代わりこの項目には自由に数式が使えます。tabular 環境では各項目で$をはさむ必要があります。

開くほうの括弧は\{と書いていることにも注意してください。一般に、{はグループ化するための文字ですから元来の意味で使うために \ を前につけています。

次の例は行列です。

表 6.5: 矢印記号

$\leftarrow$	`\leftarrow`	$\rightarrow$	`\rightarrow`	$\leftrightarrow$	`\leftrightarrow`
$\longleftarrow$	`\longleftarrow`	$\longrightarrow$	`\longrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`
$\Leftarrow$	`\Leftarrow`	$\Rightarrow$	`\Rightarrow`	$\Leftrightarrow$	`\Leftrightarrow`
$\Longleftarrow$	`\Longleftarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`
$\uparrow$	`\uparrow`	$\downarrow$	`\downarrow`	$\updownarrow$	`\updownarrow`
$\Uparrow$	`\Uparrow`	$\Downarrow$	`\Downarrow`	$\Updownarrow$	`\Updownarrow`

表 6.6: その他の記号

$\ldots$	`\ldots`	$\cdots$	`\cdots`	$\vdots$	`\vdots`	$\ddots$	`\ddots`
∞	`\infty`	$\emptyset$	`\emptyset`	ℓ	`\ell`	$\square$	`\Box`
$\bigcirc$	`\bigcirc`	$\triangle$	`\triangle`	∂	`\partial`		
$\sharp$	`\sharp`	$\flat$	`\flat`	$\natural$	`\natural`	$\angle$	`\angle`
$\spadesuit$	`\spadesuit`	$\diamondsuit$	`\diamondsuit`	$\clubsuit$	`\clubsuit`	$\heartsuit$	`\heartsuit`

次の行列

```
\[
 \left(\begin{array}{ccc}
  \cos \theta& -\sin\theta& 0\\
  \sin \theta& \cos\theta&0\\
   0  & 0  & 1  \end{array}\right)
\]
```

はz軸周りに角θ回転させる変換を与えます。

次の行列

$$\left(\begin{array}{ccc}
\cos \theta& -\sin\theta& 0\\
\sin \theta& \cos\theta&0\\
0 & 0 & 1
\end{array}\right)$$

は z 軸周りに角 θ 回転させる変換を与えます。

第7章　マクロの書き方

7.1　TEX のマクロとは

TEX のマクロとは文字列の置き換え機能です。文字列を生き返る際にパラメータを取ることが可能なので Word などにある差込印字も容易に実現できます。

また、同じような書き方をする部分をいつも同じ形式にできます。形式を直す場合にもマクロの定義の部分を直せばよいので訂正漏れがありません。

LATEX のマクロには TEX の VM が持っているもの (プリミティブコマンドといいます) とパッケージが定義するものの 2 種類があります。

7.2　マクロの書き方

LATEX のマクロは次のように `\newcommand` で定義します。

```
\newcommand{\マクロの名前}[引数の数]{マクロの定義部分}
```

マクロの名前は \ で始まり英字や (あまりお勧めしませんが) 漢字やかなの文字列です。数字や空白、記号を含めることは通常できません。またユーザーのマクロ定義と衝突しないようにするために LATEX の内部では`@` を含むマクロ名が多用されています。

引数の数は 0 個から 9 個までです。0 個の場合には `[0]` の部分は省略してかまいません。マクロの定義部分には LATEX の文が書けます。引数の参照は `#1` のように参照します。引数に名前を付けることはできません。

このテキストでは LATEX のコードとその結果が並べて書いてあります。この部分のマクロは次のようになっています。

```
\newcommand{\DisplayExampleM}[3]{%
  \begin{center}
    \begin{minipage}[t]{#2\textwidth}
       \VerbatimInput{#1.tex}
    \end{minipage}
    \hfill
    \begin{minipage}[t]{#3\textwidth}
      \input #1.tex
    \end{minipage}
  \end{center}
}
```

- このマクロの名前は\DisplayExampleM で引数は 3 つとります (1 行目)。
- 全体を中央ぞろえにするために center 環境の中 (2 行目から 10 行目) に入れています [1]。
- 二つの minipage 環境を横に並べてその間を無限に伸びる空白 (\hfill) で埋めています (6 行目)。
- minipage の幅は左のものは 2 番目引数で (3 行目)、右のものは 3 番目の引数で定義されます。ここでは単位が \textwidth としてあるので与える数値は 0.4 などとします。\textwidth は現在作成しようとしているページの割り付ける横幅です。
- 左のほうは \VerbatimInput でファイルのリストを表示し、右は単純にファイルを読み込んで組版しています。なおファイル名は拡張子なしで与えることになっていることに注意してください。

このマクロを使うとファイル名を指定するだけでソースとその結果が並べて表示できるということになります。本文のほうはそれだけ記述が簡単になり、文書の構造が見やすくなります。

7.3 マクロにおける空白について

LaTeX のマクロ内では空白または改行があると出力に微妙な差がでます。次のような点に注意してください。

- 余計な空白や行末の改行を無視するためには%で行末までを無視するようにします。これからのマクロではやたらと%が入っている例が見られますがこれがその理由です。
- 引数を与える場合には途中に空白があると無視されます。たとえば引数を 2 つとるマクロで引数が長くなった場合など次のように書いても大丈夫です。

```
\Foo{長い引数の値}
      {2 番目での引数}
```

7.4 マクロの書き方–発展

TeX には条件判断をするマクロも用意されています。すでに 3.6.3 で常に偽になる \iffalse を紹介しました。これとは逆に常に真となる \iftrue もあります。

[1] ここでの center 環境は上下に余計な空白が取られるために利用しています。

7.5 \newif

新しい判定文を作るのが \newif です。\newif\ifNewName で\ifNewName というマクロが新しく定義されます。このとき、\NewNametrue と \NewNamefalse というマクロも定義され、\NewNametrue が書かれた後では\ifNewName の後が実行され、もし、\else があればそこから \fi の部分が実行されません。\NewNamefalse が書かれた後ではこの逆の状態になります。

次の例を見てください。

```
\newif\ifFontLarge
\newcommand{\ShowMessage}{%
  \ifFontLarge
   \Large ここでは\texttt{\textbackslash Large}\else
   \large ここでは\texttt{\textbackslash large}\fi
   が有効です。}
\FontLargetrue\ShowMessage

\FontLargefalse\ShowMessage
```

ここでは \Large が有効です。
ここでは \large が有効です。

- 第 1 行目で新しい判定マクロ\ifFontLarge を定義しています。
- 同じマクロ内で記述が変わることを示すために\ShowMessage 内に\ifFontLarge を用いてフォントの大きさを変えるようにしています。
- フォントの大きさが \FontLargetrue のときは \Large に、 \FontLargefalse のときは \large になっていることを確認してください。

7.6 比較のマクロ

TEX には比較をするためのマクロとしていろいろなものがありますが、ここでは次の二つを紹介します。

- \ifx 二つのマクロの文字列が等しいときに真の部分が実行されます。
- \ifnum 二つの数の比較に用います。比較の演算子は< = >の 3 つだけです。なお、カウンターの値を使う場合には \value を用います。

\ifx の例　次の例は均等割付をするマクロの例です。TEX のマクロの変わった点が現れていますので注意してください。

```
\newcommand{\Kintou}[1]{\hfill\KintouDo#1\relax}
\newcommand{\KintouDo}[1]{%
   \ifx#1\relax\relax\else%
     #1\hfill%
     \expandafter\KintouDo\fi}
\newcommand{\MessageF}[1]{\framebox[3cm]{\Kintou{#1}}}
```

あ
あ　い
あ　い　う
あ　い　う　え
あ　い　う　え　お

```
\MessageF{あ}\\        \MessageF{あい}\\
\MessageF{あいう}\\    \MessageF{あいうえ}\\
\MessageF{あいうえお}
```

- 第 1 行目では均等割付をするマクロ\Kintou を定義しています。まずいくらでも伸びる空白 \hfill を挿入してから KintouDo というマクロを呼び出します。
- 与えられた引数のあとに \relax を付けています。これは均等割付をする文字の最後をチェックするために付けています。
- \relax は元来「何もするな」という TEX のマクロです。
- 第 2 行目からは KintouDo の定義です。このマクロは引数をひとつとりますが、Kintou あいう\relax で呼び出されると KintouDo に与えられる引数 #1 は「あ」になります。この部分が TEX の引数のとり方の特徴です。#1 の部分が与えられた文字列に置き換えられ他の後で、あたかもそこに初めからその文字列があったように処理されます。
- 3 行目で与えられた引数の値が \relax に等しいかどうか (つまり、最後の引数処理したかどうかを判定しています) を調べ、等しい場合には何もしません (二つ目の \relax)。
- 等しくない場合には引数を出力しその後に \hfill を置き、再び KintouDo を実行する、つまり再起呼び出しを行うことになります。
- KintouDo の前には \expandafter がありますが、これがないと再起呼び出しの KintouDo に渡される引数が \fi になってしまいます。
- そこで、\expandafter は直後のマクロの実行を保留し、その次のマクロを実行してから保留されたマクロを実行するということを指示します。
- これにより、\fi が実行されるので判定文が終了し、Kintou 内で渡されたその次の文字が KintouDo に渡されることになります。

やってみよう 7.1 上のマクロで\framebox[5cm]{\Kintou{{one}{two}{three}}}と \framebox[5cm]{\Kintou{one two three}}の違いを調べなさい。

\ifnum の例　次の例は縦と横の列を指定して数字を並べた表を作るマクロです。

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

```
\newcounter{TmpA}
\newcounter{TmpB}
\newcommand{\Sep}{&}
\newcommand{\MakeTable}[2]{
\setcounter{TmpB}{0}
\newcommand{\TmpAL}{#1}
\newcommand{\TmpBL}{#2}
\begin{tabular}[t]{|*{#1}{c|}}
\MakeT
\end{tabular}}
\newcommand{\MakeT}{%
    \setcounter{TmpA}{0}\MakeRow%
    \stepcounter{TmpB}%
    \ifnum\value{TmpB}<\TmpBL\\\expandafter\MakeT\fi
}
\newcommand{\MakeRow}{%
   \stepcounter{TmpA}\arabic{TmpA}%
   \ifnum\value{TmpA}=\TmpAL\else\Sep\expandafter\MakeRow\fi%
}
\MakeTable{10}{4}
```

- 行の数と列の数を管理するために TmpA と TmpB というカウンターを用意します (1 行目と 2 行目)。
- マクロの中に tabular 環境で用いられる特殊文字&を直接書くとマクロが動かないので参照するためのマクロを定義します (3 行目)
- 列を管理するカウンターの値を 0 に設定します (5 行目)。
- 表を作成するためのマクロが{\MakeTable です。このマクロは 2 つの引数をとります。この引数の値は行と列の上限値を入れるためのマクロに設定されます (6,7 行目)。
- 表の環境を定義します (8 行目から 10 行目)。表の中身はマクロ MakeT が行います。
- マクロ MakeT は 1 行分を出力するためのものです。列を管理するカウンターを初期化し (12 行目)、実際に一行分を出力するマクロ MakeRow を呼び出します。
- その後で、行を管理するカウンターを \stepcounter を用いて 1 増やし (13 行目)、上限値より小さければ (14 行目) 一行の終わりを示す\\を出力し、自分自身を呼び出します。そうでなければ終了です。
- 16 行目以降で 1 行分を構成するためのマクロが定義されています。
- カウンターの値を 1 だけ増やし、その値の算用数字にして出力します。

- 上限値に到達していなければ列の区切り文字&を出力して自分自身を呼び出します。
- 20 行目で実際に横に 10、縦に 4 行数字を書くようなマクロを実行させています。

やってみよう 7.2 単位行列を表示するマクロを書きなさい。

7.7 \ifcase

与えられた数字に応じて場合を分けるマクロが \ifcase です。\ifcase は次の形式で用いられます。

```
\ifcase<値> <0 の場合>
    \or <1 の場合>
    \or <2 の場合>
    ....
    \or <n の場合>
    \else <その他の場合>
    \fi
```

\ifcase は負の値には対処していません。

カウンターの値をアルファベットの小文字で出力するマクロは ltcounts.dtx のファイル内で

```
\ifcase#1\or a\or b\or c\or d\or e\or f\or g\or h\or i\or j\or
 k\or l\or m\or n\or o\or p\or q\or r\or s\or t\or u\or v\or w\or x\or
  y\or z\else\@ctrerr\fi}
```

のようにが \ifcase を用いて定義されています。26 より大きい場合にはエラーメッセージが出るように設定されています。

やってみよう 7.3 カウンターの値に応じて「あ」「い」...のようにひらがなが出るようなマクロを作成しなさい。

第8章　相互参照と索引の作成方法

8.1　相互参照

8.1.1　相互参照の基礎

相互参照の基礎についてはすでに 66 ページで簡単に触れました。ここではこれらのことを含めてまとめておきます。

- 参照したい場所に \label{<名前>} でマークを付けます。<名前>の部分は重複しなければ自由に付けられます。
- 参照は \ref{<名前>} で行います。ページを参照したい場合には \pageref{<名前>} で行います。
- 参照する <名前>が定義されていないとエラーが出て途中で止まります。中断 (x を入力) するかそのまま続ける (enter を入力) かを選択してください。
- 定義されていない場所には??が表示されます。
- platex にかける前に定義されてなく、今回新しく定義された場合や、ページの変更や参照する値が変わった場合にはラベルの値が変化したという次のようなメッセージが最後に出ます。

  ```
  LaTeX Warning: Label(s) may have changed. Rerun to get cross-references right.
  ```

  このときはもう一度 platex にかけてください [1]。

8.1.2　文献の参照

この文書のように末尾に参考文献を付ける場合があります。参考文献は thebibliography 環境を用います。この文書の参考文献の一部は次のようになっています。

```
\addcontentsline{toc}{chapter}{関連図書}
\begin{thebibliography}{99}
 \bibitem{BlueBook}Adobe Systems Inc.,
   Postscript Language Tutorial and Cookbook (APL), Addison-Wesley, 1985
 \bibitem{RedBook}Adobe Systems Inc.,
```

[1] まれにこれでもラベルが変更したというメッセージが出る場合があります。これは定義されていなかったラベルが定義され??の部分に数字が入ったためページの位置がずれる場合があるからです。

```
  Postscript Language Reference,  Addison-Wesley, 1999
\end{thebibliography}
```

- 標準では文献の部分は目次に登録されません。そのためね \addcontentsline を用いて追加をしています。はじめの toc は目次 (table of contents) ファイルに追加することを指示しています。次の chapter はその追加レベルが章 (chapter) であることを意味します。最後のパラメータは目次に現れる文字列になります。
- thebibliography では文献の番号の幅を指定する文字列を\begin{thebibliography}の後に記述します。ここでは{99}なので 2 桁になります。
- 各項目は \bibitem で区切ることになります。\bibitem のパラメータは本文で参照するときに使われるものになります。
- \bibitem にはオプションのパラメータをとることができます。これは

  ```
  \bibitem[Blue]{blueBook} ...
  ```

  のように宣言します。
- 本文での参照は \ref ではなく、\cite を使います。参照ページなどを記述したい場合には \cite[20 ページ]{blueBook}のように指定します。
- オプションのパラメータがある場合にはオプションのパラメータが参照部分にも文献リストの方にも現れます。
- パラメータがない場合には番号で現れます (本書と同じ)。

8.1.3　相互参照で表示される値について

相互参照で表示される値はその環境で指定されているカウンターの \the<カウンター名>で出力の値になります。このコマンドに飾りを付けるなどすると次のように困った現象がおきます。

```
\renewcommand{\thesubsection}{%
  \fbox{\thesection.\arabic{subsection}}%
  \hspace{1em}}
  \subsection
   [Bad Section Format(文書とは無関係です)]
   {Bad Section Format}
\label{badsecformat}
この項を参照すると \ref{badsecformat}
となり参照場所にも枠がつきます。
```

8.1.4　Bad Section Format

この項を参照すると 8.1.4 となり参照場所にも枠がつきます。

この作用は目次にも影響を与えていることに注意してください。

これを避けるためには次のように \@seccntformat を変更します。なお、このコマンドでは \chapter には影響を与えません。

```
\makeatletter
\renewcommand{\@seccntformat}[1]{%
\fbox{\csname the#1\endcsname}\hspace{1em}}
\makeatother
\subsection
[Good Section Format(文書とは無関係です)]
{Good Section Format}
\label{goodsectionformat}
この項を参照すると \ref{goodsectionformat}
となり参照場所にも枠がつきません。
```

8.1.5 Good Section Format

この項を参照すると 8.1.5 となり参照場所にも枠がつきません。

- 通常、@ はマクロ名に使用できないようになっています。一時的に使用できるようにするために \makeatletter を使用します。
- 節番号の表示形式を定義している \@seccntformat を再定義します。
- 定義し終わったら@を再びマクロ名に含ますことができないように \makeatother を使用します。
- このコマンドは与えられたカウンターを引数とするのでその名前を含んだ \the マクロを定義するために \csname を用いています。\endcsname までの範囲がマクロの名称になります。ここで \@seccntformat{subsection} とすると結果は\thesubsection\hspace{1em}となります。
- 目次のほうも参照しているのと同じ形式になっていることを確認してください。

また、自分で指定したカウンターの値を利用したい場合にはそのカウンタを使うはじめのところで \refstepcounter{<カウンター名>} と指定します。

```
\newcounter{mycnt}
\newcommand{\ShowMe}{\refstepcounter{mycnt}\arabic{mycnt}}
\begin{center}
 \begin{tabular}[t]{|c|c|c|c|}
  \hline
 \ShowMe& \ShowMe\label{No2}& \ShowMe& \ShowMe\\ \hline
 \ShowMe& \ShowMe&  \ShowMe&  \ShowMe\label{No}\\ \hline
 \end{tabular}
\end{center}
ここで \ref{No2} と \ref{No} が参照できます。
```

1	2	3	4
5	6	7	8

ここで 2 と 8 が参照できます。

- まず新しいカウンターを宣言します (1 行目)。
- 毎回同じ記述が続くのでカウンターの値を 1 増やし、表示するマクロを定義します (2 行目)。
- 表のなかでカウンターの値を増やしながらその値の表を作成しています。

- 2 番目と 8 番目にラベルを付けていてそれを下で参照しています。そのときの値が参照されていることを確認してください。

やってみよう 8.1 上記の例で\ShowMe の定義のなかで\refstepcounter を\stepcounter に変えたらどうなるか調べなさい。

8.2 索引の作成

8.2.1 索引作成の基礎

索引の作成は LaTeX だけではできません。次のように行います。
まず、プリアンブルに \makeindex と記述してください。

文書内にマークを付ける 索引に載せたい文字が現れるところに \index{<用語>} と書きます。この<用語>は文書内に現れません。ある項目の下の項目としたい場合には\index{<見出し用語>!<下位の用語>}のように記述します。この指定は 3 段階まで可能です。このようにした文書 (foo.tex とします) を LaTeX にかけると foo.idx というファイルができます。

索引の並べ替え 出来上がったファイルを mendex というプログラムで処理します。

```
mendex foo
```

これにより同一の用語のページ番号がまとめられ、索引の見出しを挿入した foo.ind ファイルが作成されます。
元の文書の最後に \printindex を書き入れてもう一度 LaTeX に通します。これで索引が作成できます。

8.2.2 索引作成の詳細

読みを付ける 上記の方法では日本語の索引は残念ながらできません。mendex が日本語の読みがわからないので処理できないというエラーを出します。\index{ 読みがな@<用語>} というように@で区切った前に読みがなを付けます。したがって下位の項目の場合には上位の項目の読みもつける必要があるので次のようになり、原稿は見にくくなります。

```
\index{すうしき@数式!ぶんすう@分数}
```

なお、英字の文字列でも、表示するものと読みの位置が異なる場合には読みを与える必要があります。この文書で LaTeX のマクロに対して \ を取り除いたものの位置に置くようにしています。これは次のような \index を作成しているからです。

```
\index{section@\texttt{\textbackslash section}}
```

この文書では索引生成のためにマクロを定義して、読みは別のプログラムで付け、その後で mendex のプログラムに通して最終的な処理をさせています。

索引の体裁を変える mendex が作成する索引の体裁を変えることができます。この文書の索引を決める体裁は次のような内容です。

```
1  % Make index with lower character
2  preamble "\\begin{theindex}\\small\n"
3  %
4  % leading letter is a little large
5  lethead_prefix        "\\item {\\large\\bf "
6  lethead_suffix        "}\\nopagebreak[4]"
7  %
8  % Japanese leading letter is hiragana
9  lethead_flag 1
10 letter_head 2
11 %
12 % set delimeters between item and page numbers as \dotfill .or
13 delim_0        " \\dotfill "
14 delim_1        " \\dotfill "
```

- 2 行目では索引をはじめる先頭のコマンドを指定します。\\ となっているのは文字列内での \ を記述するためのものです。
  ここでは索引を小文字にするように指定しています。
- 5 行目では見出し文字の体裁を定義しています。ここでは\large にして、太文字 (\bf) にしています。
- 6 行目では見出し文字を入れた後の処理をする部分を記述しています。見出し文字の直後にページが変わらないように\nopagebreak[4] を入れています。[4] を指定するとページがここでもっとも変わりにくくなります。
- 9 行目では見出し文字を出力すること指定しています。この値を負にするとアルファベットの見出しが小文字になります。
- 10 行目で日本語の見出し文字をひらがなに指定しています。0 や 1 ではカタカナになります。
- 13 行目と 14 行目では索引に載った語とページ数の間に入れる文字列を指定しています。ここでは \dotfill なのでピリオドがいくつか入って与えられた幅の左端に索引語が、右にページ数が載ることになります。\dotfill は \hfill のピリオド版です。

このファイルを mystyle.sty としたとき

```
mendex -s mystyle.sty foo
```

とするとこのスタイルで索引が作成されます。

mendex の詳しい仕様は<TeX のインストール場所>/share/texmf/doc/mendexk/base にある mendex.pdf を見てください。mendex 自身が読みを辞書から付けるという機能持っていますが、この機能についての解説もこの文書にあります。

やってみよう 8.2 索引のスタイルパラメータを変化させて見出しの文字がまったくない場合やカタカナになる索引の体裁ができることを確認しなさい。

ページ番号の書体を場所によって変える　この文書ではしていませんが、索引に載ったページの書体を変更することが可能です。たとえば、定義が出ているところを太文字にすることで他のページとの差別化が図れます。これらの機能をまとめると次のようになります。

表 8.1: \index の中のオプション

記述	機能
|textbf	文字を太文字にします。
|(	項目の記述開始場所を指定します。終了位置の指定との間のページが – で結ばれます。
|(textbf	項目の開始の指定と太文字の指定です。順序に注意してください。
|)	項目の記述終了位置を指定します。
|see{別の項目}	「別の項目」への参照を示します。ページは出力されません。

8.3 hyperref パッケージ

hyperref パッケージの概要と使い方　この文書では参照の部分や索引の項目のページの部分の色が変わっていて、そこをクリックすると該当の場所にジャンプできます。また、一部のところでは外部の URL へジャンプもできます。このような機能は hyperref パッケージを用いることで実現されています。

このパッケージを用いるためにはプリアンブルで

```
\usepackage[dvipdfm,
        bookmarks=true,bookmarksnumbered=true,
        bookmarksopen=true]{hyperref}
```

を書いておきます。オプションの引数として処理するためのドライバーが指定できます。ここでは標準のドライバであるは hypertex です。さらに、bookmarks を true にして、しおりを作ることを指定し、bookmarksnumbere を true にして、しおりの表題の前に番号をつけています。また、この設定では pdf ファイルが開かれたときにしおりも表示されるようになっています。しおりを開かないようにするには pdfpagemode を none に指定します。

URL へのリンク　URL へのリンクを張るためには

```
\href{URL}{テキスト}
```

とします。これで本文には [テキスト」部分が現れ、プレビューワでは URL へのリンクが生成されます。この文書では

```
\newcommand{\Href}[2]{\href{#1}{#2}\footnote{\texttt{#1}}}
```

というマクロを作成して脚注に URL が見えるようにしています。

やってみよう 8.3 hyperref パッケージを使った場合と使わない場合で .idx ファイルがどのように変わるか調べなさい。

\autoref について なお、hyperref パッケージでは \autoref で参照も可能です。図\ref{graphx2+1}(図 5.2) と書く代わりに \autoref{graphx2+1}(図 5.2) と書きます。これによりリンク先の文字列も「図」の部分まで含まれるようになります。

なお、[10, p.46] にはこの「図」に相当する文字列を定義してあるマクロのリストがありますがこれはそもそも日本語対応になっていないパッケージなので英語で表示されてしまいます。hyperref.sty の中には次のような記述があるので\def の後にあるマクロ名で定義するとうまくいきます [2]。

```
\def\equationautorefname{Equation}%
\def\footnoteautorefname{footnote}%
\def\itemautorefname{item}%
\def\figureautorefname{Figure}%
\def\tableautorefname{Table}%
\def\partautorefname{Part}%
\def\appendixautorefname{Appendix}%
\def\chapterautorefname{chapter}%
\def\sectionautorefname{section}%
\def\subsectionautorefname{subsection}%
\def\subsubsectionautorefname{subsubsection}%
\def\paragraphautorefname{paragraph}%
\def\subparagraphautorefname{subparagraph}%
\def\FancyVerbLineautorefname{line}%
\def\theoremautorefname{Theorem}%
```

本文との体裁を合わせるのであれば

```
\renewcommand{\figureautorefname}{\figurename}
```

などとするのがよいでしょう。

しおりの作成など 上記の設定で pdf ファイルに変換するとしおりが作成されますが、日本語がうまく表示できません。これはしおりの部分の文字コードを Unicode にする必要があるからです。このためには次のような処理が必要です。

[2] \def は TEX においてマクロを定義するためのコマンドです。\newcommand で定義してもできますが、この方法では定義できない引数の与え方もできるので場合によっては使うのもよいでしょう。

```
platex foo
platex foo
out2uni foo
platex foo
dvipdfm foo
```

hyperref パッケージは pdf ファイルに対する命令を記述できる機能があります。詳しくは [10] を見てください。

8.4　この文書での索引の作成方法

この文書では自作のマクロとそれにより生成された.idx ファイルを処理するプログラムを作成して索引が少し作りやすいようにしています。

- \keyitem{用語} や\keysubitem{下位の用語}{上位の用語}などのマクロを定義して繰り返しが少なくなるようにしています。「用語」、「下位の用語」は本文に表示されます。
- これらのマクロを基礎に、カテゴリーごと (LaTeX のパッケージや環境) に簡単な索引が付けられるようなマクロを作成しています。
- 辞書を自作して perl で.idx ファイルを処理をして読みを付け、一部の用語ではこの処理中に索引の項目に対して項目の後に LaTeX のマクロも付加しています。
- 上位と下位の項目では下位の項目に現れる上位項目の部分が—— で表示するようになっていますが、これもここの処理部分で行っています。
- これらの処理を行ったものに対して mendex が処理を行っています。

LaTeX ファイルは原則としてすべてテキストファイルなのでこのようなことが可能になっています。

これらのマクロはこの文書を作成するために新たに開発したのではなく、筆者のいろいろな文書ですでに使用しているものを一部修正して利用しています。

第9章　LaTeXでプレゼンテーションをする

9.1　LaTeXとプレゼンテーション

プレゼンテーションというとほとんどの人が Microsoft 社の PowerPoint で作成するでしょう。LaTeX の歴史では初めからプレゼンテーションにも対応することが考えられていました。プレゼンテーションの歴史は透明な用紙に下から光を当てて画面に表示する OHP(Over Head Projector) から始まりました。この用紙は A4 縦の程度の大きさでしたから、LaTeX でも大きなフォントを用いてプレゼンテーションの原稿を作成する `slides` というクラスがあります。このクラスではフォントの大きさが少し足りないので大き目のフォントサイズを定義した `foils` というパッケージもあります。見た目には PowerPoint ほどカラフルにできないのであまり使う人はいないようです。

ここでは比較的最近開発された `LaTeX-beamer`[12] を紹介します。

9.2　`beamer` の特徴

beamer の特徴としては次のことが挙げられます。

- プレゼンテーションとしてプロジェクタを想定してます。したがって、デフォルトの出力用紙は横長です。
- プレゼンテーションは全画面表示をすることを想定しているの実際に作られる画面の大きさはかなり小さくなっています。このアイディアにより大きいサイズのフォントを用意しなくて済むようになっているようです。
- 複数の画面を配布用にまとめるハンドアウト用のオプションがあります。
- アニメーションの機能は同じページを複数生成することで実現しています。ハンドアウトのときは最後のページが表示されます。
- 標準でいくつかのテーマが用意されていて、見かけがかなり異なるプレゼンテーションが作成できます。
- 全体のシートを、outer, inner, font, color の 4 つの構成部分にわけ、それぞれを組み合わせてテーマを構成しています。

最近の配布では標準で含まれています。

9.3 原稿の構成

原稿は通常の LaTeX 文書と同様に次のような部分から構成されます。

9.3.1 プリアンブル

次のリストはプリアンブルの例です。

```
1 \documentclass[dvips]{beamer}
2 \usepackage{pgfpages, graphicx, xcolor}
3 \usepackage{moreverb}
4 \usetheme{AnnArbor}
5 \title{\LaTeX を用いてプレゼンテーション資料を作成しよう}
6 \author{自分の名前を!}
7 \institute{自分の所属を}
8 \date{平成 21 年 4 月 15 日}
```

- 1 行目の \documentclass に beamer を用いることを宣言しています。オプションに dvipsk を指定することで、出来上がった dvi ファイルを dvips を用いて PDF ファイルに変換できるようになります。
- 2 行目と 3 行目でで使用するパッケージを定義しています。いつもこれらのパッケージを使うものと思ってください。
- 4 行目の \usetheme は各プレゼンテーションの様式を決めるものです。

  beamer をインストールしたフォルダの themes/theme の下に定義のファイルがあります。いろいろな形式のものがありますので好きなものを選んでください。ここでは配色が少し派手な AnnArbor を選んでいます。
- 5 行目から 7 行目ではそれぞれプレゼンテーションのタイトル (\title)、講演者の名前 (\author)、講演者の所属 (\institute) を定義しています。
- 8 行目は日付を定義しています (\date)。これを定義しないときは LaTeX にかけた日つまり当日の日が設定されます。内容を指定しないときは日付の表示がまったくないことになります。

9.3.2 本文

プリアンブルの後に通常の LaTeX の文書と同様に、\begin{document} と \end{document} の間に本文を書きます。

各シートの構成

各シートは frame 環境か \frame を用います。通常、各シートにはタイトルを付けます。簡略な方法もありますが、ここでは明示的に示す \frametitle と \framesubtitle を紹介しておきます。

タイトルページと目次

タイトルページは \titlepage を用います。これをシートに埋め込むので \frame{\titlepage} と記述すれば十分です。また、通常の文書と同じように \tableofcontents を利用すれば文書内の \section や \subsection を拾って目次が作成されます。この目次にはページ番号が付きません。

9.4 プレゼンテーションファイルの作成法

出来上がったソースファイルからプレゼンテーション資料を作成するためには次のようなコマンドを実行します。beamer の機能を利用するためには dviout では十分に対応していないので、pdf ファイルに変換することになります。そのためにいくつかのコマンドを実行する必要があります。(最後の 2 行は一つのコマンドなので改行しないこと)。

```
platex ex
platex ex
dvips -z -f ex.dvi | bkmk2uni > ex.ps
gswin32c -dDEVICEWIDTHPOINTS=363 -dDEVICEHEIGHTPOINTS=272
       -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=ex.pdf ex.ps
```

一連のコマンドを実行するために次のようなバッチファイルを用意すると良いでしょう。

```
platex %1
if %ERRORLEVEL% GTR 0 GOTO END
platex %1
dvipsk -z -f %1.dvi | bkmk2uni > %1.ps
if %ERRORLEVEL% GTR 0 GOTO END
gswin32c -dDEVICEWIDTHPOINTS=363 -dDEVICEHEIGHTPOINTS=272 ^
       -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=%1.pdf %1.ps
:END
```

このファイル名を mk.bat とししたとき、foo.tex のファイルをプレゼンテーション用のファイルに変換するためには次のコマンドを実行します (処理するファイルの拡張子を付けないこと！)。

```
mk.bat foo
```

cygwin のような環境で make を利用したい場合には次のような Makefile を用意します。

```
TARGET        = example
TARGETH        = example-handout
LATEX        = platex
DVIPS        = dvipsk -Ppdf -z
GS        = gswin32c -dNOPAUSE -dBATCH -sDEVICE=pdfwrite

SRC        = $(TARGET).tex
```

```
SRCH        = $(TARGETH).tex
COMMON        = $(TARGET)-body.tex

all: $(TARGET).pdf $(TARGETH).pdf
$(TARGET).pdf: $(SRC) $(COMMON)
        $(LATEX) $(TARGET)
        $(LATEX) $(TARGET)
        $(DVIPS) -f $(TARGET).dvi | bkmk2uni > $(TARGET).ps
        $(GS) -dDEVICEWIDTHPOINTS=363 -dDEVICEHEIGHTPOINTS=272 \\
              -sOutputFile=$(TARGET).pdf $(TARGET).ps
```

これにより make でファイルが作成できることになります。

9.4.1 ハンドアウトの作成

ハンドアウトとはプレゼンテーションの各シートをいくつかまとめて 1 ページに印刷したものです。ハンドアウトを作成する \documentclass には \handout オプションをつけます。また、1 ページに入れるページ数は \pgfpagesuselayout で指定します。

次の例は 1 ページに 8 枚入れるものを作成するためのコマンドです。

```
\pgfpagesuselayout{8 on 1}[border shrink=5mm]
```

- デフォルトは用紙いっぱいに割り付けられるので端を空けるためのオプションの引数を指定をする
- 1 ページに 2 枚、4 枚が指定できる。枚数に応じて縦横は自動で変わります。

したがって、プリアンブルを二つ用意し、片方はプレゼンテーション用、残りはハンドアウト用とし、本文を共通のファイルとしてそれを読み込むようにするのが便利です。

次の例は情報メディア特別講義で作成したスライドの原稿のものです。

プレゼンテーション用ファイル

```
\documentclass[dvipsk]{beamer}
\usepackage{pgfpages}
\input whattex-beamer.tex
```

ハンドアウト用ファイル

```
\documentclass[dvipsk,handout]{beamer}
\usepackage{pgfpages}
\pgfpagesuselayout{8 on 1}[a4j,border shrink=3mm]
\input whattex-beamer.tex
```

ハンドアウトのほうには紙の大きさの指定も入っていることに注意してください。

9.5 ページの体裁の変更

デフォルトのままでは日本語のフォントが読みにくかったりします。テーマから探すのが面倒であったり一部だけ変えるようなコマンドがあります。

```
1 \renewcommand{\familydefault}{\sfdefault}
2 \renewcommand{\kanjifamilydefault}{\gtdefault}
3 \setbeamerfont{title}{size=\large,series=\bfseries}
4 \setbeamerfont{frametitle}{size=\large,series=\bfseries}
5 \setbeamertemplate{frametitle}[default][center]
```

- 1 行目はフォントファミリのデフォルトを変えています。
- 2 行目は幹事のフォントファミリをゴシックの標準に設定しています。
- \setbeamerfont はスライドを構成する部署のフォントを変更するものです。ここでは 3 行目でタイトルの、4 行目でフレームタイトルのフォントの大きさとファミリをそれぞれ変えています。
- 5 行目はフレームタイトルの表示位置を中央にしています。

9.6 オーバーレイの指定

オーバーレイとはプレゼンテーションの画面の一部を見えなくしていく手法です。余り込み入ったことをすると見ている人の思考を妨げますので使い図偽に注意しましょう。

9.6.1 箇条書きの項目を順番に表示する

次の例は箇条書きの項目を順番に表示するものです。
この部分は次のようになっています。

```
1 \begin{frame}
2 \frametitle{オーバーレイの例}
3 \begin{itemize}[<+-| alert@+>]
4  \item 1 番目です
5  \item 2 番目です
6  \item 3 番目です
7 \end{itemize}
8 \end{frame}
```

itemize 環境の後ろにオプションのパラメータ [<+-| alert@+>] がついています。初めにある+は各\item の上からの位置を示し、-はその番号以降を意味します。|以降の部分はそれを修飾するものです。ここでは alert@+となっているのでその順番の時には alert(強調) の形式で表示するよ

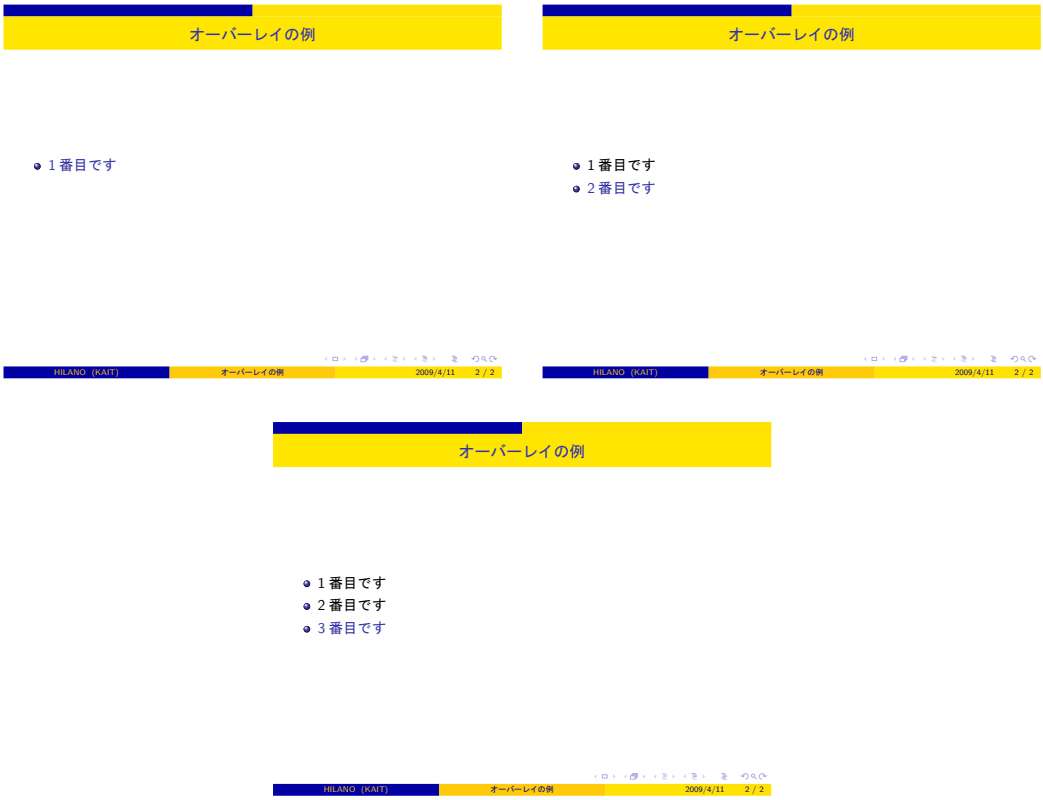


図 9.1: 箇条書きの項目を順番に表示

うに指定しています。上の図でも 1 番目の項目は 1 枚目だけ色が変わっています。テーマによってこの色は変わります。

これと同じことは次のようにしてもできます。

```
1 \begin{frame}
2 \frametitle{オーバーレイの例}
3 \begin{itemize}
4  \item<1-> \alert<1>{1 番目です}
5  \item<2-> \alert<2>{2 番目です}
6  \item<3-> \alert<3>{3 番目です}
7 \end{itemize}
8 \end{frame}
```

この方法では\item の後ろに表示のパラメータを記述します。したがって表す順序を変えたいときなどは便利な方法ですが、途中に項目を付け加えると修正が面倒になります。\alert のあとのパラメータも同じように付け加えることができます。

やってみよう 9.1 次のような場合にどのように表示されるか確認しなさい。

```
\begin{frame}
\frametitle{オーバーレイの例}
\begin{itemize}
 \item<1-2> \alert<2>{1 番目です}
 \item<2-3> \alert<-2>{2 番目です}
 \item \alert<1,3>{3 番目です}
\end{itemize}
\end{frame}
```

一つの項目を 2 回以上に分けて表示するためにはこのような方法は取れません。そのためには \onslide を用いると良いでしょう。

```
\begin{frame}
\frametitle{オーバーレイの例}
\begin{itemize}
 \item \onslide<1->\alert<1>{1 番目です}

       \onslide<2->1 番目のコメントです。
 \item \onslide<3->\alert<3>{2 番目です}

       \onslide<4->2 番目のコメントです。
 \item \onslide<5->\alert<5>{3 番目です}

       \onslide<6->3 番目のコメントです。
 \item \onslide ここはずっと表示されています。
\end{itemize}
\end{frame}
```

この例では一つの\item が 2 回に分けて表示されます。範囲は次の\onslide がくるまでです。最後の\item は\onslide に表示のパラメータを与えていないので常に表示されます。

このほかにもやり方はあるようですがここでは解説しません。

付 録A　PSファイル

次のリストは図 5.2で利用した PS ファイルのリストです。

```
%!PS-Adobe-3.0 EPSF-3.0
%%BoundingBox: 27 27 171 230
%%Title: C:\usr\local\home\tel\seiseki\2004\yamaguchi\graphx2+1.ps
%%Creator: GSview from C:\usr\local\home\tel\...\graphx2+1.ps
%%CreationDate: Sun Apr 25 16:47:09 2004
%%Pages: 1
%%EndComments
%%Page: 1 1
%%BeginDocument: C:\usr\local\home\tel\...\graphx2+1.ps
/arrow {
    /endy exch def /endx exch def
    /starty exch def /startx exch def
    gsave
    startx starty translate
    /dx endx startx sub def
    /dy endy starty sub def
    dx dx mul dy dy mul add sqrt
    dup dx exch div /costheta exch def
        dy exch div /sintheta exch def
        /arrowlength currentlinewidth 5 mul def
        /nx dx costheta arrowlength mul sub def
        /ny dy sintheta arrowlength mul sub def
        0 0 moveto nx ny lineto stroke
        nx ny translate
        /newarrowwidth 1.5 currentlinewidth mul def
        newarrowwidth sintheta mul newarrowwidth neg costheta mul moveto
        arrowlength costheta mul arrowlength sintheta mul lineto
        newarrowwidth sintheta neg mul newarrowwidth costheta mul lineto
        closepath fill
    grestore
} def
```

```
/func{dup mul 10 div 10 add} def

72 25.4 div dup scale
0.4 setlinewidth

/xH 30 def   /yH 60 def
/xL -10 def  /yL 0 def
/xLL xL -5 add def /yLL yL -5 add def
/xHH xH  5 add def /yHH yH  5 add def

/RmClipLine {
    /yHHT exch  1 add def /xHHT exch  1 add def
    /yLLT exch -1 add def /xLLT exch -1 add def
    setgray
    xLLT yLLT moveto xLLT yHHT lineto
    xHHT yHHT lineto xHHT yLLT lineto
    closepath
} def

xLL neg 10 add yLL neg 10 add translate

xLL 0 xHH 0 arrow 0 yLL 0 yHH arrow

gsave
  1 xLL yLL xHH yHH RmClipLine clip
  0 setgray
  0.3 setlinewidth
  -10 20 moveto
  -10 1 40 {
    dup func lineto
  } for stroke
  0.2 setlinewidth

  10 dup func 20 sub exch 10 sub exch moveto
  10 dup func exch 25 add exch 50 add lineto stroke
grestore
1 setlinewidth 1 xLL yLL xHH yHH RmClipLine stroke
showpage
%%EndDocument
%%Trailer
```

付 録B　LaTeXの参考書について

LaTeX に関する情報は英語で書かれたものがやはり多いです。日本語の環境では異なることがあるので国内の関連図書も参考にするとよいでしょう。ここに挙げた英語の文献はすべて日本語訳が出ています。ただし、[8] は 2004 年に第 2 版が出ていて分量が初版に比べて倍以上になっています。この訳はまだ出ていないようです。

B.1　LaTeX の入門書

LaTeX の入門書としてはなかなかよい日本語の本がありません。強いてあげるのであれば [6] がよいでしょう。手っ取り早く学ぶのであれば [11] がよいでしょう。日本語に訳されたものもありますが元になった版が古いので日本語版と見比べて見るのもよいかもしれません。

インストール用の CD もついてくる [14] は初歩から書いてあるようにも見受けられますが内容がすぐに難しくなるのであまりお勧めではありません。少し慣れたら [8] を手元においてどのようなパッケージがあるかの参考にするのがよいでしょう。図を含む文書を作成したい場合には [9] を参考にしてください。この内容のうち一部は [8] へ移動したようです。また、LaTeX の文書を pdf や html 文書に直すことを考えているのならば [10] を参考にしてください。

なお、2004 年に上記の LaTeXCompanion 3 部作に [7] が加わり、導入部分の記述はここに移されたようです。この本の日本語訳はまだ出ていません。これに伴い他の 3 冊も内容に大幅な変化が見受けられますので注意してください。

本書で取り上げたプレゼンテーション用の文書クラスである LaTeX-beamer については [9] にかなりまとまった解説があります。また、日本語で書かれたサイトも多く見受けられます。正式な解説書は配布されたパッケージの中にあります [12]。200 ページを超えるものですが、途中にはプレゼンテーション資料はどのように作成するのかという解説もありなかなか読み応えがあります。本人が作成したプレゼンテーションもありますのでそれを動かしてみて面白そうなのがあったらソースを見て研究するのも良いでしょう。

B.2　TeX のマクロの書き方の参考書

第 7 で一部紹介したように TeX のマクロは非常に癖がある書き方を要求されます。繰り返しなどを含まないものであればそれほど難しいことはないと思います。

[13] では LaTeX の内部に関することがらにも触れながらかなり突っ込んだマクロの書き方が解説されています。また、TeX の解説書ですが [3] や Knuth 自身の書籍 [4] も非常に参考になります。

B.3 PostScript の参考書

図を PostScript で書く場合には [1] が最適です。日本語訳も出ていますが今では入手が困難なようです。言語のすべてを知りたければ [2] を見るしかありません。Lebel 1 のときは薄かったこの本も Lebel 3 は巨大な厚さになっています。Adobe 社の Web サイトにも pdf で置いてあります [2]。

関連図書

[1] Adobe Systems Inc., Postscript Language Tutorial and Cookbook (APL), Addison-Wesley, 1985

[2] Adobe Systems Inc., Postscript Language Reference third ed., Adobe Systems Incorporated `http://www.adobe.com/products/postscript/pdfs/PLRM.pdf`

[3] Victor Eijkhout, TEX by Topic, A TEXnician's Reference, Addison-Wesley, 1991

[4] Donald E. Knuth, The TEXBook, Computers & Typesetting / A, American Mathematical Society , Adison-Wesley publishing Company, 1996

[5] Donale E. Knuth, The Art of Computer Programming, Vol. 1 Fundamental Algorithms, Third Ed., Adison-Wesley publishing Company, 1997

[6] Leslie Lamport, LATEX: A Documentation Preparation System User's Guide and Reference Manual, Addison-Wesley Series on Tools and Techniques for Computer Typesetting, 1994

[7] Helmut Kopka, Patrick W. Daly, Guide to LATEX, 4th ed., Addison-Wesley Series on Tools and Techniques for Computer Typesetting, 2004

[8] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, The LATEX Companion, Addison-Wesley Series on Tools and Techniques for Computer Typesetting, 2004/4/14

[9] Michel Goossens, Sebastian Rahtz, Frank Mittelbach, The LATEX Graphics Companion: Illustrating Documents With TEX and Postscript, Addison-Wesley Series on Tools and Techniques for Computer Typesetting, 1997/4/15

[10] Michel Goossens, S. P. Q. Rahtz, The LATEX Web Companion: Integrating TEX, Html and Xml, Addison-Wesley Series on Tools and Techniques for Computer Typesetting, 1999

[11] Tobias Oetiker, Hubert Partl, Irene Hyna and Elisabeth Schlegl, The Not So Short Introduction to LATEX 2_ε Or LATEX 2_ε in 139 minutes, version 4.20 2006, `http://www.ring.gr.jp/archives/text/CTAN/info/lshort/english/`

[12] Till Tautau, The BEAMER *class* Manual for version 3.07

[13] ページ・エンタープライゼズ (株), LATEX2_ε マクロ&クラス プログラミング基礎解説, 技術評論社, 2002

[14] 奥村 晴彦, ［改訂第 3 版］LATEX 2_ε 美文書作成入門, 技術評論社, 2004

索引

Symbols

\@seccntformat 76, 77
\[..... 65
\$ 26
% 37
& 41
\{ 26
\} 26
\\ 26, 41

A

\addcontentsline 76
\addtocounter 35
\alert 88
\Alph 36
\alph 36
\alpha(α) 67
\angle($\angle$) 68
\arabic 35, 36
\author 29, 84
\autoref 81

B

beamer 84
beamer パッケージ .
    \alert 88
    \framesubtitle 84
    \frametitle 84
    handout 86
    \onslide 89
    \pgfpagesuselayout 86
    \setbeamerfont 87
\beta(β) 67
\bfsereis 28
\bibitem 76
\bigcirc($\bigcirc$) 47, 68
bookmarks 80
bookmarksnumbere 80
BoundingBox 61
\Box($\square$) 68
\bullet($\bullet$) 67

C

\cap($\cap$) 67
\caption 48, 50, 60
\cdot($\cdot$) 67
\cdots 65
\cdots($\cdots$) 68
\chapter 76
\chi(χ) 67
\circ($\circ$) 67
\circle 54
\circle* 54
\cite 76
\cleardoublepage 34
\clearpage 34
\cline 45, 49
\clubsuit($\clubsuit$) 68
\copyright (©) 39
\csname 77
CTAN 4
\cup($\cup$) 67

D

\date 29, 84
\ddots($\ddots$) 68
\Delta(Δ) 67
\delta(δ) 67
\dfrac 63
\diamondsuit($\diamondsuit$) 68
\displaystyle 63
\div($\div$) 67
\documentclass 25, 28, 29, 37, 84, 86
\dotfill 79
\Downarrow($\Downarrow$) 68
\downarrow($\downarrow$) 68
dvi 84
dvipsk 84

E

\ell(ℓ) 68
\else 71
\em 28
\emph 28
\emptyset($\emptyset$) 68
\endcsname 77
\endfirsthead 50
\endfoot 50
\endhead 50
\endlastfoot 50
\enumi 36

\enumii 36
\enumiii 36
\enumiv 36
\epsilon(ε) 67
\equiv($\equiv$) 67
\eta(η) 67
\expandafter 72

F

\fbox 38
\fi 37, 71, 72
\fill 37, 38
\flat($\flat$) 68
\fnsymbol 36
\footnote 31
\footnotesize 27
\frac 63, 64
\frame 84, 85
\framebox 38
\framesubtitle 84
\frametitle 84

G

\Gamma(Γ) 67
\gamma(γ) 67
\geq($\geq$) 67
\geqq($\geqq$) 67
GhostScript 12

H

\handout 86
\headheight 37
\headsep 37
\heartsuit($\heartsuit$) 68
\hfill 37, 38, 60, 70, 72, 79
\hline 41, 45, 49
\hoffset 37
\hspace 37, 38
\hspace* 37
\Huge 27
\huge 27
hypertex 80

I

\ifcase 74
\iffalse 37, 70
\ifnum 71, 72
\iftrue 70
\ifx 71
\in($\in$) 67
\include 36
\includegraphics 57–59, 61
\includeonly 36
\index 78
\infty(∞) 68
\input 36
\institute 84
\iota(ι) 67
\item 30
\itshape 28

J

jbook 25
jpeg2ps.exe 60
jpeg ファイルを PostScript ファイルに変換する (jpeg2ps.exe) 60
jreport 25

K

\kappa(κ) 67

L

\label 66, 75
\Lambda(Λ) 67
\lambda(λ) 67
landscape 25
\LARGE 27
\Large 27
\large 27
\LaTeX 26
latex2html 5
\ldots($\ldots$) 68
\Leftarrow($\Leftarrow$) 68
\leftarrow($\leftarrow$) 68
\Leftrightarrow($\Leftrightarrow$) 68
\leftrightarrow($\leftrightarrow$) 68
\leq($\leq$) 67
\leqq($\leqq$) 67
\line 53
\linethickness 56
\Longleftarrow($\Longleftarrow$) 68
\longleftarrow($\longleftarrow$) 68
\Longleftrightarrow($\Longleftrightarrow$) 68
\longleftrightarrow($\longleftrightarrow$) 68
\Longrightarrow($\Longrightarrow$) 68
\longrightarrow($\longrightarrow$) 68

M

\makeatletter 77
\makeatother 77
\makebox 53, 54
\makeindex 78
\maketitle 29
\mdsereis 28
mendex 78, 79
\mp($\mp$) 67
\mu(μ) 67
\multicolumn 46
\multiput 55, 56
\multirow 47

N

\natural(♮) 68
\neq(≠) 67
\newcommand 69
\newcounter 36
\newif 71
\newpage 34
\newtheorem 29, 34, 35
\ni(∋) 67
\noindent 26
\nonumber 66
\normalfont 28
\normalsize 27
\nu(ν) 67

O

\oddsidemargin 37
\Omega(Ω) 67
\omega(ω) 67
\onslide 89

P

\pageref 66, 75
\paragraph 29
\parbox 38
\partial(∂) 68
pdfpagemode 80
\pgfpagesuselayout 86
\Phi(Φ) 67
\phi(ϕ) 67
\Pi(Π) 67
\pi(π) 67
\pm(±) 67
PostScript 11
\pounds (£) 39
\printindex 78
\Psi(Ψ) 67
\psi(ψ) 67
\put 53, 55, 58, 59

R

\ref 66, 75, 76
\refstepcounter 77
\relax 72
\renewcommand 35
\rho(ρ) 67
\Rightarrow(⇒) 68
\rightarrow(→) 68
ring サーバー 7, 12
\rmfamily 28
\Roman 36
\roman 35, 36
\rotatebox 58
\rule 45

S

\scalebox 58, 59, 61
\scriptsize 27
\scshape 28
\section 85
\setbeamerfont 87
\setcounter 35
\setlength 53
\sffamily 28
\sharp(♯) 68
\Sigma(Σ) 67
\sigma(σ) 67
\sim(∼) 67
slides 83
\slshape 28
\small 27
SmartDoc 5
\spadesuit(♠) 68
\sqrt($\sqrt{}$) 65
\stepcounter 73
\stretch 38
\subsection 85
\subset(⊂) 67
\subseteq(⊆) 67
\subsubsection 29
\supset(⊃) 67
\supseteq(⊇) 67

T

\tableofcontents 29, 85
\tableoffigure 60
\tau(τ) 67
\textasciicircum 26
\textbackslash 26
\textbf 28
\textcent (¢) 39
\textdollar ($) 39
\textdollor 26
\texteuro (€) 39
\textheight 37
\textit 28
\textmd 28
\textnormal 28
\textparagraph (¶) 39
\textperthousand (‰) 39
\textregistered (®) 39
\textrm 28
\textsc 28
\textsection (§) 39
\textsf 28
\textsl 28
\texttrademark (™) 39
\texttt 28
\textunderscore 26

\textup 28
\textwidth 37, 38, 70
\textyen (¥) 39
\thepage 35
\thesection 36
\Theta(Θ) 67
\theta(θ) 67
\thinlines 56
\times(×) 67
\tiny 27
\title 29, 84
\titlepage 85
\today 31
\topmargin 37
\triangle(△) 68
\ttfamily 28

U

\unitlength 53
\Uparrow(⇑) 68
\uparrow(↑) 68
\Updownarrow(⇕) 68
\updownarrow(↕) 68
\upshape 28
\Upsilon(Υ) 67
\upsilon(υ) 67
URL へのリンク 80
\usepaakage 49
\usepackage 29, 47
\usetheme 84

V

\value 35, 71
\varepsilon(ε) 67
\varphi(φ) 67
\varpi(ϖ) 67
\varsigma(ς) 67
\vartheta(ϑ) 67
\vdots(⋮) 68
\VerbatimInput 33, 70
\voffset 37
\vspace 38

X

\Xi(Ξ) 67
\xi(ξ) 67

Z

\zeta(ζ) 67

い

印刷範囲 37

う

上付き 64

お

オーバーレイ 87

か

カウンター 35
—— の値の表示を変える
(\arabic,\roman,\Roman) 35, 36
—— の値を変える (\setcounter) 35
—— の値を指定した値だけ変化
(\addtocounter) 35
—— を作る (\newcounter) 36
箇条書き (itemize 環境) 30
カレントディレクトリ 8
環境
align 66
align* 66
array 67
atable 60
center 30, 48, 60, 70
document 25, 84
enumerate 30
equation 66
figure 34, 59, 60
flushiright 31
frame 84
itemize 30, 87
listing 32
longtable 49
minipage 31, 70
picture 53, 54, 58, 59
table 34, 42, 48
tabular 38, 41, 42, 48–50, 67, 73
thebibliography 75, 76
Verbatim 32–34
verbatim 32
Verbatim* 33
環境を定義する (\newtheorem) 34

き

脚注 (\footnote) 31
今日の日付を表示する (\today) 31
ギリシャ文字 66
均等割付 37, 71

く

空白 26
縦方向の—— (\vspace) 38
無限に伸び縮みができる (\hfill) 37
グループ化 ({,}) 26

こ

コメント 26

さ

差込印字 69

参照
番号による—— (\ref) 66
ページによる—— (\pageref) 66

し
下付き 64
条件判断 (\if,\ifx,\ifcase) 70–72, 74

す
図目次 (\listoffigure) 60

そ
添え字 64

た
段落の始まりに空白を入れない (\noindent) .. 26

ち
中央ぞろえ (center 環境) 30

て
ディメンションレジスタ 37
印刷範囲 37

な
長さの単位
\fill 37
\headheight 37
\headsep 37
\hoffset 37
\oddsidemargin 37
\stretch 38
\textheight 37
\textwidth 37, 70
\topmargin 37
\unitlength 53
\voffset 37

は
パッケージ
amsmath 66
amssymb 66
array 43, 44, 48
fancyvrb 32
foils 5, 83
graphicx 56, 57
hyperref 22, 80–82
LaTeX-beamer 83
longtable 49
moreverb 32
multirow 47
supertab 49, 51
textcomp 39

ふ
ファイルを取り込む (\input) 36
プリアンブル 25
プレビューア 2
プログラムリストの表示 32
番号を続ける (listingcont 環境) 32
文書クラス 2, 25
分数の書き方 (\frac,\dfrac) 64

へ
ページ内に小さなページを作成
(minipage 環境) 31

ほ
ボックス
段落 (\parbox) 38
枠付きの—— (\fbox,\framebox) 38

ま
マクロ 26, 69
—— の名前 69
—— の引数の数 69

み
右ぞろえ (flushright 環境) 31

も
目次の出力 (\tableofcontents) 29

よ
横向き (landscape) 25

ら
ラベル (\label) 66