LaTeX 2ε科技排版指南

邓建松　彭冉冉　陈长松　编著

科学出版社
2001

内 容 简 介

本书依据国内外关于TEX与LATEX(尤其是当前最新版本的LATEX 2ε)的资料,并结合编著者多年来使用该排版软件的经验,详细讲解了LATEX 2ε这一功能强大的科技文献排版软件。本书既完整地介绍了中英文LATEX 2ε所有排版命令,又对其编程功能做了讲述。

无论是初学者,还是经验丰富的TEX用户,本书都是应用LATEX 2ε排科技文章的最佳参考书。

图书在版编目(CIP)数据

LATEX 2ε科技排版指南/邓建松,彭冉冉,陈长松编著 . -北京:科学出版社,2001.9

ISBN 7-03-009239-2

Ⅰ.L… Ⅱ.①邓… ②彭… ③陈… Ⅲ. 排版-应用软件,LATEX

Ⅳ.TS803.23

中国版本图书馆 CIP 数据核字(2001)第 10090 号

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

丽源印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2001 年 9 月第 一 版 开本:787×1092 1/16

2001 年 9 月第一次印刷 印张:24

印数:1—3 000 字数:545 000

定价: 40.00 元

(如有印装质量问题,我社负责调换〈北燕〉)

前　言

编著者第一次接触 TEX 和 LATEX 大约是 10 年前的事情了，当时可用的资料只有一些油印的小册子。那时，编著者还是一名本科生，没有进行任何科研工作，因此也就没有对它发生多大的兴趣。

6 年前编著者开始读研究生，需要自己录入排版，但是这时首先选择的排版软件是Microsoft Word(版本 5.0)，因它具有 "所见即所得" 的良好性能。当时编著者用它录入了大量的文章，甚至成本的教材。直到有一天，编辑说用 Word 排版的数学公式太难看了，这时才知道，在文章的排版中，要想得到标准而漂亮的数学公式，并不是一件简单的事情。从这时起编著者才真正地开始学习和使用 TEX 和 LATEX.

从初学者的角度来看，TEX 和 LATEX 实在是太难掌握了，因为别的不说，单是每开始排版一篇文章时，LATEX 文档开头的导言部分就使人满头雾水。而各种名称的 TEX 格式 (如 Plain TEX, $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX, emTEX, LATEX, $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX) 就更是令人眼花缭乱了。另外，初学者在用 TEX 排版文章时，经常会出错，那些难以理解的出错信息，也实在令人却步。

TEX 是一种文本处理系统，它是美国 Stanford 大学的 Donald Knuth 于 1977 年 5 月开始着手设计的。几年后，Leslie Lamport 开发了 LATEX, 它以 TEX 作为自己的格式处理基础。LATEX 的诞生，对 TEX 的普及起了相当重要的作用。发展到今天，LATEX 已逐步成为了主流，因此本书中主要介绍的是 LATEX, 而且重点是它的最新版本 LATEX 2_ε. 我们详细地介绍了如何应用 LATEX 2_ε 进行科技文章和书籍的排版。

我们编写本书的主要目的就是帮助初学者快速掌握 LATEX, 但实际上即使对已经用过相当长时间的 LATEX 用户，本书也不失为一本内容齐全的参考书。在编写本书的时候，我们参考了国内外出版的各种 TEX 和 LATEX 使用手册和资料 (在参考文献中列出了这些资料的详细目录)。另外，由于美国数学会推出的 $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX 具有相当丰富的数学符号，并且可以排出更复杂的数学公式，因此我们在第五章中也同时介绍 $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX 的数学排版功能。

全书共分四部分，下面我们简要列出本书的主要内容。

第一部分为 **基本 LATEX**, 这一部分共包含两章，其中第一章简要介绍 TEX 和 LATEX 的历史，并列出了编辑 LATEX 文档的两种推荐编辑器 EditPlus 和 WinEdt. 在第二章中对利用 LATEX 排版文章进行快速简略的介绍，主要列出了 LATEX 命令的组成、各种字符的输入方法和长度的定义，并给出了排版一篇完整文章的示例。这个示例中包含了 20 多个典型公式的样例，这组样例对后面进一步学习公式的排版相当有帮助，在阅读第五章时也不妨参考一下。

第二部分为 **高级 LATEX**, 详细完整地介绍了所有 LATEX 排版功能。第三章介绍页面组织的各项功能，其中包括对文档类的简要介绍，列出了类的样式选项，以及在相应的文档类中可以出现的章节命令及结构。接着介绍生成目录表、插图和表格清单的方法，最后说明如何对文本中不恰当的地方进行细微调整，并解释了控制断词的命令和方法。

第四章对文本排版的各种环境和声明进行了详细介绍。列举了改变字体大小和属性的命令，给出了排版居中和缩进文本块的方法。本章中不但介绍了 LaTeX 预定义的各种列表环境，如 enumerate, itemize 和 description 等，而且给出了广义列表的语法，用这种列表可以构造出前一节的所有列表，而且还可以生成其他各种特色的列表环境。另外，还说明了在文章中如何使用定理结构、制表位和盒子命令。本章最后的几节介绍了显示源文本，添加注释和标记脚注与边注的方法。

第五章介绍的是数学公式的排版，当初之所以开发 TeX 和 LaTeX 系统，就是为了排版出标准而规范的数学公式。首先讲解了数学环境的构造，然后逐一说明了公式的组成部分，列出了获得各种数学符号的命令，并说明了公式中其他要素的构造方法，例如给公式加编号，多行公式的编辑等。在介绍前面内容的过程中，我们同时列出了 $\mathcal{AMS}$-LaTeX 中对 LaTeX 数学排版增强的部分，特别地，我们在 5.6 节中介绍了 $\mathcal{AMS}$-LaTeX 为排版多行公式而增加的新环境和功能。本章的最后一节说明如何对数学公式进行精细调整。

第六章中讲解了 LaTeX 中插图和表格的编辑方法。我们在前几节中介绍的是生成插图的命令，接着解释了生成表格的环境和参数，最后给出了 LaTeX 文档中浮动对象的处理方法。

第七章中给出了在 LaTeX 中定义命令和环境的方法。为此首先介绍了 LaTeX 记数器和长度，然后详细介绍了在 LaTeX 如何自定义无参数、有参数以及有可省参数的命令和环境。

第八章中介绍的则是比较复杂的功能，例如多文件的组合、正文中的引用、参考文献的组织、新字体选择框架的启用，以及书信和幻灯片的编辑等，另外还介绍了如何插入外部图形的方法。

在第三部分，我们介绍的是 **CCT 中文 LaTeX 系统**，这一部分内容主要参考了 CCT 开发者提供的使用手册。首先在第九章介绍了各种中文 TeX 系统，并说明了 CCT 中文 LaTeX 的构成。在第十章则详细列举了 CCT 中文 LaTeX 系统新增的命令和功能。

本书最后一部分是长达 100 多页的 **附录**，内容包含了各种符号的列表，以及高级用户可能会感兴趣的编辑参考文献数据库的方法和 LaTeX 程序设计命令，为了帮助用户在使用 LaTeX 时应付各种出错和警告信息，我们也列出了针对各种信息的处理方法，在附录中还介绍了计算机现代字体，并在附录的最后给出了索引，以方便用户检索各种命令和概念。

为了方便 LaTeX 爱好者交流经验，共享资料，编著者在 1999 年 4 月建立了 TeXGuru 站点，当前网址为 http://202.38.68.78/~texguru/，因此编著者在这里要感谢中国科学技术大学数学系科学计算与计算机图形学实验室，如果没有该实验室的良好软硬件环境，本书是不可能这么快完成的。另外，科学出版社的杨波先生对本书的出版起了非常大的作用，作者在此向他表示衷心的感谢。

邓建松

2000 年 8 月于中国科学技术大学

第一部分　基本 LaTeX

第一章　简　介

在当今时代，计算机最通用的一个功能就是对文本的电子化处理，这一过程主要由如下四步组成：

1. 文本输入到计算机里，存贮供以后修改；
2. 把输入文本格式化，即用长度相同的行和特定尺寸的页排出来；
3. 在计算机的显示器上显示查看格式化后的结果；
4. 把最终的输出送到打印机上打印出来。

有很多文字处理系统 (例如， MS Word) 可以在一个软件包中同时实现这四方面的功能，因此用户也就意识不到上面的步骤划分。而且，第 3,4 步实际上是相同的，都是把格式化后的结果送到一个输出设备上。

TeX 是一种文本格式化程序，它只进行第 2 步的处理。任何一种文本编辑器都可以进行第 1 步的操作，即输入和修改源文本。而字处理程序就不一定满足这里的要求，因为这种程序通常在文件中加入很多不可见的控制字符。对字处理程序而言，“所见即所得”是一个非常好的功能。但是另一方面，有的时候，我们更看重排版结果的优美与规范，以及系统的稳定性。因为谁也不想自己辛辛苦苦编辑出来的结果，由于系统崩溃而毁于一旦。

用编辑器生成的文件，若要用作格式化程序的输入，其中应该包含一些特殊命令，这些命令是用可以看到的普通文本表示的。从某种意义上来说，这种供格式化用的命令集合很像一种包装语言，它只是用来表示段落、章节等等从哪儿开始，而不是直接对文本进行格式化处理。而在用程序进行格式化的过程中，这些指令是如何被解释的，则要看所选用的版面设计格式而定。同样的文本，在不同的版式下可能形成完全不同的式样。

格式化程序的功能远不止这些。实际上 TeX 也是一种有丰富功能的编程语言，有经验的用户可以用它编写代码，以增加某些功能。 LaTeX 自身也就是一组复杂的宏集合。任何用户都可以通过编程，或者直接利用其他程序员已设计好的宏对 LaTeX 进行扩展。TeX 和 LaTeX 的功能并不只限于包含在基本安装版本中的那些内容。

对于格式化软件而言，文本处理的最后一步是把结果送到输出设备上，这里的输出设备可以是打印机、计算机显示器，甚至文件。实现这种转化的软件称为驱动程序；它

把格式化程序已编码好的输出翻译成用户可以使用的某一设备上的特定指令。这也就是说，对每种类型的打印机，也就必须有相应的驱动程序。

1.1 TEX 及其历史

在所有的可以排版科技著作的计算机格式化程序中，功能最强的就要属 Stanford 大学的 Donald E. Knuth 所设计的 TEX 程序了，其名字是由在数学公式中经常用的希腊字母 $\tau\epsilon\chi$ 的大写形式组成。正是由于这个原因， TEX 最后一个字母的发音并不是 [ks]，而是 [k]。

除了 TEX 外，Knuth 还设计了另一个软件 METAFONT， 用来生成各种字符字体。在标准的 TEX 软件包中有 75 种不同设计尺寸的字体，而且每种字体有八种不同的放缩比例。所有这些字体都是用 METAFONT程序生成的。为了满足其他应用的需要，还设计了其他字符字体，如古斯拉夫语和日语字母的字体，利用这些字体，就可以把相应文本按书籍质量要求排版。

1.1.1 TEX 程序

最基本的 TEX 程序由一些很初等的命令组成，它们可以完成简单的排版操作和程序设计功能。然而，在 TEX 中还可以用这些初等命令定义一些复杂的高级命令。这样就可以利用低级的结构块，形成一个用户界面相当友好的环境。

当运行 TEX 时，该程序首先读取格式文件 (格式文件中包含各种以基本语言写成的高级命令，也包含分割单词的连字号安排模式)，接着就处理源文件 (源文件由要处理的真正文本，以及在格式文件中已有定义的各种命令组成)。

创建新格式也是一件需要由知识丰富的程序员来做的事情。把定义写到一个源文件中，这个文件接着被一个名叫 initex 的特殊版本的 TEX 程序处理。它采用一种紧凑的方式存贮这些新格式，以利于通常的 TEX 程序很快地读取。

1.1.2 Plain TEX

Knuth 设计了一个名叫 Plain TEX 的基本格式，以便与低层次的 TEX 对应。这种格式是 TEX 排版的相当基本的部分，以致于我们有时候根本分不清到底哪是真正的 TEX 处理程序，哪是这个特殊的格式。大多数声称只使用 TEX 的人，实际上指的是只用 Plain TEX。

Plain TEX 也是其他高级格式的基础，这些格式的广泛应用进一步加深了人们把 TEX 和 Plain TEX 认为是同一事物的错觉。

1.1.3 LATEX

Plain TEX 的重点还只是停留在如何排版的层次上，而不是从一位作者的角度来看问题。当然对 TEX 深层功能的进一步发掘，需要相当高超的编程技巧。因此它的进一步应用就需要高级排版和程序设计人员的参加。

正是由于这种原因，美国计算机学家 Leslie Lamport 开发了 LaTeX 格式，这种格式提供了一组生成复杂文档所需要的更高级命令。利用这种格式，即使使用者没有排版和程序设计的知识也可以充分发挥由 TeX 所提供的强大功能，能在几天，甚至几小时内生成大量具有书籍印刷质量的结果。在生成复杂表格和数学公式方面，这一点表现得尤为突出。

LaTeX 相对于其基础 Plain TeX 而言，更像一个包装语言。它可以在作者根本不知道所以然的情况下，自动给出标题、章节、表格目录、交叉索引、公式编号、文献引用和浮动图表。版面布局信息包含在类文件中，这些类文件并不是位于源文件中，使用者不但可以直接套用这些布局，还可以进行修改。

LaTeX 是在 20 世纪 80 年代出现的，同其他软件一样，它也周期性地进行更新和修订。

1.1.4 LaTeX 2ε

由于 LaTeX 相当普及，以及它在许多原本没有想像到的领域中的扩展，再加上计算机技术的日新月异，特别是价格低廉而功能强大的激光打印机的出现，使得相当广泛的一类格式都冠以 LaTeX 的标签。为了尝试建立一个真正的改进标准，在 1989 年 Leslie Lamport, Frank Mittelbach, Chris Rowley 和 Rainer Schöpf 创立了 LaTeX3 项目。他们的目标是建立一个最优的、有效的命令集合，这些命令均来源于为了实现某一目的而设计的各种宏包。

正如项目名称所表明的，它的目标就是得到 LaTeX 的一个新版本 3。朝向这个长期目标迈进的第一步就是在 1994 年中发行了 LaTeX 2ε 并出版了 Lamport 基本手册第二版。LaTeX 2ε 是在令人瞩目的 LaTeX3 出现之前的现行标准版本。

实际上，在 LaTeX 2ε 出现之前，其处理字体的安装和选择的一些部分已经以新字体选择框架 (或 NFSS) 的形式公开了，而且被许多组织和个人集成到其软件中。这种框架有两个版本，但不幸的是，这两个分别相应于 LaTeX2.09 和 LaTeX 2ε 的版本并不兼容。后来以一种与 2.09 版本完全兼容的方式对 NFSS 进行了重新实现。

1.2 LaTeX 2ε的新内容

本节内容主要针对于那些已相当熟悉 LaTeX2.09 的读者。下面简要列出了 LaTeX 2ε 的新功能。如果读者是第一次接触 LaTeX, 完全可以略过该节内容。

1.2.1 类与宏包

LaTeX2.09 与 LaTeX 之间的一个最本质的差别就在于文档的第一条命令，这条命令声明了文档的布局信息。

在 LaTeX2.09 中，必须像下面这样来声明所需要的主样式，这个样式同时带有一些选项：

```
\documentstyle[ifthen,12pt,titlepage]{article}
```

这里的主样式是 article，它保存在一个叫 article.sty 的文件中，而同时用 12pt 作为基本字体大小，titlepage 规定把标题单独放在一页上。但是由于主样式并不理解 ifthen 选项，所以会自动读入名为 ifthen.sty 的文件。这种处理未定义选项的方法是把 LaTeX 的扩展或补充代码读入文档中的主要途径。类似于 ifthen.sty 的附加文件有很多，都可以用来做为样式选项（我们也称之为子样式）。

然而，在 LaTeX 中这些补充选项与真正的内部选项之间是有显著差别的。因此现在主样式更名为类，而附加文件则被称为宏包。上面那个初始化声明就变为

```
\documentclass[12pt,titlepage]{article}
\usepackage{ifthen}
```

这里的布局信息包含在 article.cls 文件中，它可以处理 12pt 和 titlepage 选项。而 ifthen.sty 文件还像以前那样读取，但是它也可以有自己的内部选项。不仅如此，那些列在 \documentclass 命令中的选项，都被看作是全局的，因此对所有宏包都有作用（见 3.1.2 节）。

初始化声明 \documentstyle 在 LaTeX 2ε 中仍可以使用，这时 LaTeX 2ε 会切换到一个兼容模式，来模拟 LaTeX2.09 的行为。

为了帮助那些坚韧不拔的 LaTeX 程序员更好地进行程序设计，LaTeX 2ε 增加了许多新功能。它对选项的处理做了改进，还可以在宏包加进选项；增加了一些安全机制，以保证版本号的匹配；提供了更好的测试方法，以保证在读取其他文件时，如果该文件不存在，可以采取其他的措施。在附录 C 中对这些程序设计要素进行了描述。

1.2.2 字体管理

在 LaTeX2.09 中，TeX 的计算机现代字体 (computer modern font) 被牢靠地固定在格式中。在人们喜欢用的字体也就那么几种的年代里，这不失为一种可行的方法。但到了今天，由于可用的字体数目繁多，尤其是 Post Script 打印机的出现，就非常需要设计一个有弹性的系统。新字体选择框架 (NFSS) 应运而生，并与 LaTeX 2ε 结合为一体。要选择非计算机现代字体作为基本字体只需要一些很简单的重定义（见 8.5 节）。

NFSS 也改变了在文档内部引进字体的方法。LaTeX2.09 继承了 TeX 的字体命令，如 \bf(黑体)、\it(斜体) 都只能选择某一特定的字体。这些命令中只有字体大小维持不变。而在 NFSS 中，字体是用某种属性来描述的，可以彼此独立地进行选择。因此就有可能先选择黑体，然后选斜体，从而得到黑斜体，而这在 LaTeX2.09 中是行不通的。

LaTeX 2ε 鼓励使用字体选择命令，不提倡使用字体声明。例如，为了强调某一文字，命令 \emph{hello} 就比声明 {\em hello} 要好。这样的命令对初学者来说，更符合逻辑，虽然习惯于用后者的经验丰富的 LaTeX2.09 用户可能持相左的看法。

在数学模式中，文本的字体是通过特殊的数学字母命令来选择的，而不是原来的字体声明。也就是说，在数学模式中不允许使用原来的 \rm, \bf 和 \cal 等声明，而代之以 \mathrm, \mathbf 和 \mathcal 等有参数的命令。

1.2.3 浮动对象的安排

在 LaTeX2.09 中，一个令人头痛的问题就是如何安排浮动对象 (图与表)，使它出现在人们最希望看到的地方。然而对浮动对象的安排有一套相当复杂的规则，并不是所有的人都能很好地掌握。LaTeX 2ε 提供了两种新的机制，以控制这一过程，其中一种不鼓励对浮动对象的安排，另一种则鼓励这种安排。

通过使用 `\suppressfloats` 命令可以使当前页中没有浮动对象。作为选项，可以使用参数 `t` 或 `b` 来只禁止浮动对象不出现在当前页面的顶部或底部。

另一方面，有一个新的浮动位置指定符 `!`，如果在浮动对象的定位参数中使用了这个符号，那么它可以取消所在页对可以出现的浮动对象的数目与文本总量的限制。这也克服了通常会出现的一种页面不满的情形。而原来要解决这个问题，就必须重定义 `\textfraction` 或其他浮动安排参数，并进行无数次的调试才有可能成功。浮动位置指定符 `!` 的使用方法与其他参数一样。例如，

```
\begin{figure}[!]
```

现在为了使浮动对象与文本更好的分离开，也可以在其顶部或底部画上定义好的标尺。在 6.7 节中对这些功能进行了介绍。

1.2.4 扩充的语法

与 2.09 版本相比，LaTeX 2ε 对几条命令的语法进行了扩充。当然原来的语法仍然有效。

- `\newcommand`, `\renewcommand`, `\newenvironment` 和 `\renewenvironment` 命令用来定义新的命令和环境，这些命令除了几个必须的参数外，还可以包含一个可省略的参数 (7.3 节和 7.4 节)。
- 盒子命令 `\makebox`, `\framebox` 和 `\savebox` 在定义其实际尺寸时，可以引用其自然的尺度。也就是说，你可以用它来定义宽度为自然宽度两倍的盒子。见 4.7 节。
- 现在的 `\parbox` 命令和 `minipage` 环境除了需要定义水平宽度外，还可以指定一个竖直高度。同时提供了一个新的内部安排参数来决定盒子中的文本是否需要被推向顶部或底部，或者居中安置，甚至可以伸展文本以填满整个盒子 (4.7.5 节)。
- 以前的 `\settowidth` 命令可以用来测量某些文本的宽度;现在又补充了 `\settoheight` 和 `\settodepth` 命令 (7.2 节)。

1.2.5 与 LaTeX2.09 的兼容性

为了使 LaTeX 2ε 与 LaTeX2.09 尽可能地保持兼容性，系统设计人员已做了各种各样的努力。这样就可以使根据原来版本标准写的文档，在新的版本下可以得到相同的结果。同时宏包中绝大多数的样式 (或子样式) 选项具有与以前相同的功能，而不需任何修正。

这只是最理想的情形。实际上，两种版本之间一定存在着不兼容之处。那么用户在哪些情形中会遇到这种不兼容现象呢?

- 如果只使用高级命令，在命令中不包含 @ 字符，那么兼容性是 100%。
- 如果使用了内部命令，就可能会导致问题，特别是用到了盒子命令或者输出程序。

- 如果使用了内部字体控制命令，特别是那些来自于 lfonts.tex 文件的命令，就很有可能会出现问题。然而，应该指出的是，即使这样，也比 NFSS 的第一个版本与第二个版本之间的不兼容性小。

据目前的经验，我们发现，即使把盒子命令和输出程序算在内，也只有很少几个宏包不能在 LaTeX 2ε 下运行。我们遇到的最糟糕情形是，在 LaTeX2.09 下调入一个支持文件，而这个文件在 LaTeX 2ε 下却不一定存在。如果该文件被强行调入，就会出现严重错误的信息。

1.2.6 版本的判别

如果你不知道在你所用的系统中是否安装了 LaTeX 2ε，有许多方法可以检查。在任一 LaTeX 作业开始处，都会在显示器和抄本文件 (transcript file) 中列出格式的名字和日期:

如果用的是 LaTeX2.09，信息为 `LaTeX Version 2.09 <25 March 1992>`，

如果用的是 LaTeX 2ε，信息为 `LaTeX2e <1996/12/01>`，

当然你的日期可以与这里的不同。

如果不喜欢这种方法，还可以尝试处理一个文件，第一行用

```
\documentclass.
```

如果你得到如下的错误信息:

```
! Undefined command sequence
      \documentclass
```

那么你用的就不是 LaTeX 2ε。

1.2.7 更新 LaTeX 2ε

按计划在每年的 6 月和 12 月要对 LaTeX 进行更新，不仅要改进内部编码，还要增加一些额外的功能。这就是说，随着时间的推移，仅仅只说一个文档是 LaTeX 2ε 格式的还不够，不但要指明必要的版本，还要加上格式发行的日期。

为此我们需要在文档文件中声明一个最早的可以处理文档所用全部功能的版本。在靠近文档开头的地方，加上一条鉴别命令 (C.2.1 节)。如果用的是第一个 LaTeX 2ε 版本，那就应该加上

```
\NeedsTeXFormat{LaTeX2e}[1994/06/01]
```

这里的日期必须是数字，格式为年 / 月 / 日，年月日中间用斜线分开，且须加上必要的零。如果有这一命令的文件被更早的 LaTeX 版本处理，就会显示一条警告信息。

1.3 如何使用本书

本书是教科书与参考手册的混合体。它解释了所有的 LaTeX 以及 CCT emTeX 的基本组成，特别是那些新标准的 LaTeX 2ε 中的部分。本书适用于那些经验很少，甚至没有计算机使用经验的用户。

书中的内容是逐步加深的，在第一部分与第二部分中有些内容是重复的。因此读者在刚开始学习的时候，没有必要强求掌握新出现的所有概念。

1.3.1 字体约定

为了更好地理解本书所述内容，我们建议读者要首先熟悉一下从 2.2~2.5 节开始出现的那种描述命令或环境语法的方式。

在描述命令语法时，对那些必须精确照原样输入的部分用打字机字体 (如 \begin) 表示，而那些可以改变的部分或者文本自身，则用斜体 (相应于英文符号) 或楷体 (相应于中文) 表示。例如，宏包 graphpap 中生成格纸的命令就用如下方式陈述：

\graphpaper[数](*x*,*y*)(*lx*,*ly*)

用打字机字体表示的部分是不能省略的，而 数 表示必须在此处加上格线间隔的单位数。而 x, y, lx, ly 分别表示格式的左下角坐标以及格式的尺寸。

在正文中，有时为了表示强调，我们也用楷体表示特定的术语或规则，例如：“公式中有时会包含一串点 ...，表示 等等”。

1.4 LaTeX的编辑

我们在前面已说过，编辑 LaTeX 可以采用任何文本编辑器，但前提条件是这个编辑器生成的文件必须是没有格式的，即它不会向文件中增加任何不可见的控制字符。目前非常流行的 Microsoft Word 是一个非常优秀的编辑器，用它排版没有数学公式或者只有少量公式的普通文章或办公信函，是非常便捷的，因为它具有 “所见即所得” 的强大优势。但是这个编辑器并不满足我们这里的要求。

在微机上，如果所用操作系统为 DOS 或 Windows，那么最经常用的文本编辑器是 Edit。但是采用这个编辑器时，我们不得不频繁进出编辑器，以编译和查看排版结果。因此我们期望能有一个集成式的编辑环境。我们要求这个环境，一方面具有丰富的编辑功能，另一方面，不用退出编辑器，就可以马上知道排版结果。EditPlus 和 Winedt 是我们的首选。

1.4.1 EditPlus

EditPlus 文本编辑器是 ES Computing 公司提供的一个 32 位的 Windows 程序，可以用它替换 Windows 附件中 Notepad 编辑器，它也为网页的编辑与设计提供了许多强有力的工具。

这个编辑器可以按照系统预定的语法文件或者用户自定义的语法文件，用彩色显示出某种类型文件中的保留词或命令。例如，若正在编辑 C++ 源程序，那么当输入完 float 的最后一个字母时，这个单词就会以彩色显示，从而提示你的输入是正确的。

另外这个编辑器还允许用户利用外部程序设置一些工具，而且有自动补足的功能。正是由于这个编辑器具有这些强大功能，我们才选择它做为 LaTeX 的编辑工具。首先我们创建出相应于 LaTeX 类型的语法文件和自动补足文件，然后对 EditPlus 进行一系列的配置或

直接修改注册表文件(配置或修改的具体过程以及所需的相关文件请见http://202.38.68.78/~texguru/ 站点)。在做了上述配置或修改以后，现在编辑 LaTeX 文件时，就有很多便利的地方，一方面，所有的 LaTeX 命令在输入完毕后，都会自动变色(要保证在命令的前后与其他命令之间有适当的分隔，如空格)，甚至 TeX， LaTeX 和 $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX 的命令所具有的颜色是不同的，从而提示你包含必需的宏包文件。在使用环境时，编辑器会自动补足 \end{...} 部分。例如，当输入完

```
\begin{equation
```

后，我们不要输入另一半 }，只要按空格，那么你就会得到

```
\begin{equation}
\label{}


\end{equation}
```

光标自动停留在 \label{} 的大括号内，让你输入一个关键词，以供引用。由于我们使用 equation 环境，就是为了对这个公式进行引用，否则用的就是 $$...$$ 环境了。注意，这里千万不要在输入 equation 后的 } 再按空格，那样是不会有这种自动补足效果的。

另外，我们还可以利用 EditPlus 自动生成 LaTeX 模板，因为初学者对 LaTeX 导言部分的组成是非常头疼的。利用模板，就可以很容易地着手编辑正文。下面就是我们自己创建的 LaTeX 文件模板。

```
\documentclass[11pt]{article}
%\documentclass[11pt]{book}
%\usepackage{graphicx} %We can use any other package if necessary
%\usepackage{amsmath} % If you want to use AMS-LaTeX, comment out these
%\usepackage{amssymb} % two commands.
%\usepackage{chemsym}

\setlength{\parindent}{12pt}
\setlength{\parskip}{3pt plus1pt minus2pt}
\setlength{\baselineskip}{20pt plus2pt minus1pt}
\setlength{\textheight}{21true cm}
\setlength{\textwidth}{14.5true cm}

\title{Thesis}
\author{Deng Jiansong\\texguru@263.net}

\begin{document}%

%Create title of papers.
```

```
\maketitle

%Edit the contents as you want.
%If you want to insert some PostScript pictures, use command:
%\includegraphics{xxx.eps}

\end{document}%
```

只要在 EditPlus 文件中，从菜单上选择 New ，然后选你要创建的文件类型 (CCT emTeX 或 LaTeXe)，就可以得到不同类型的模板。把这个文件保存成后缀为 .tex(相应于英文 LaTeX) 或 .ctx(相应于 CCT 中文 LaTeX) 的文件，然后就可以输入和编译。在上述模板中，我们在许多行前面加上了注释符号 %, 这样在实际使用时，就可以通过去掉某些行前面的注释符号，或者在某些行前面加上注释符号，而改变所使用的缺省设置。例如，可以把

```
\documentclass[11pt]{article}
%\documentclass[11pt]{book}
```

变为

```
%\documentclass[11pt]{article}
\documentclass[11pt]{book}
```

这样文档使用的就是 book.cls 类，而不是 article.cls 类。

1.4.2 WinEdt

WinEdt32 是由 Aleksander Simonic 开发的一个快速、强大的编辑器，它具有相当高的弹性，可以在 Win95, Win98 和 WinNT 等系统中使用。该系统是用 Borland Delphi 开发的 32 位应用程序。

可以用 WinEdt 编辑相当庞大的文本文件。可以同时打开多个文件，遵从通常多文档界面 (MDI) 的约定，利用窗口菜单或 "Ctrl+Tab" 按键在文件中来回切换。除了一些标准的编辑功能外， WinEdt32 还支持：

- 块 / 列选择和操作，关键词高亮显示，拼写检查并显示出拼写有误的单词，定界符自动匹配，恢复桌面，宏功能，用户自定义图形界面， ....
- WinEdt 可以搜集文档中的特定信息 (例如目录表或者标签)，以便于快速移动和索引。

WinEdt 的弹性表现在

* 用户可完全自定义菜单 (包括快捷键) 的组成;
* 可完全自定义工具条;
* 用户可自定义与不同上下文有关的弹出菜单，当单击鼠标右键时会出现这些菜单。

我们可以对 WinEdt 进行配置，以适用于不同的需要。不过，它特别适合于编写 TeX 文档，特别地，它提供了一组模板，要输入 TeX 和 LaTeX 甚至 $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX 符号，只要在这个模板的对应符号处单击一下就可以输入该符号，这大大简少了记忆符号名称的工

作量。不过，相对于 EditPlus 而言，我们至今还没有找到显示行号的功能，这也不能不说是一种遗憾。在 WinEdt 中我们可以安装 DOS 和 Windows 程序，然后只要利用菜单命令或者工具条按钮就可以激活这个程序。而且， WinEdt 可以把当前打开的文件做为参数传递给这个程序。在定义了一些工具 (例如 TEX, LATEX, BIBTEX等) 后，与编写 TEX 文档有关的绝大多数任务都只需要单击一个工具条按钮或者选择一条菜单命令就可以完成。

WinEdt 起初是与 MikTEX 一同使用的，为了使其支持中文 LATEX，我们也可以对其进行一些配置，具体详情见 TEXGuru 主页 (`http://202.38.68.78/~texguru/`)。

第二章　LaTeX排版入门

LaTeX 是目前国际上流行的排版软件，它特别适合于科技文章和书籍的编辑和排版。与目前流行的 WPS 和 Word 软件相比，它在字符质量、排版功能和数学公式的处理方面均胜一筹。目前，许多著名的期刊，如 SIAM, AMS, Phy. Rev. 等都接受 LaTeX 格式的稿件，甚至有许多杂志、期刊有自己的类与宏包，以便对文章做出满意的排版。

在本章简单介绍 LaTeX 排版的基本概念，并演示一些结构的使用方法。

2.1　LaTeX 文件的基础知识

2.1.1　文本与命令

所有的文本文件都由字符组成，几个字符放在一起组成单词，由多个单词组成句子，句子再组成段落，而段落可以做为更大单位 (如章节) 的一部分。

单词由一个或多个字符组成，由空格或回车终止。TeX 把空格和回车都做为单词的结束符，而单词间的空格多少是无关紧要的，即单词的间隔与空格的数目无关。

段落是由一个或多个空行分隔的，同样，段落间隔与空行的数目无关。TeX 把段落中的所有单词当做一个单词长串来处理，选择单词间隔，以使得尽可能地均匀，而且每一行都要向左或向右调整 (对齐)。行的断开是自动的，与文本的输入基本无关。

行的间隔与所选的字体尺寸有关。段落是以首行的缩进或者行间隔的增大为标志的。在必要的时候，段和行的间隔都会发生细微的变化。TeX(和 LaTeX) 通过调整这些间隔，使得一页的顶部和底部位于需要的位置。可以在文档内改变这个位置。当断行结束后，会自动地进行分页。

对于最简单的情形，即文本文件中只有输入文本，TeX 就会用标准的宽度和高度处理这一文本。也就是说，把要排版的文本均匀分成行、段、页。

然而，通常的 LaTeX 文档不仅仅是由要处理的文本组成，还要包含规定处理文本方式的命令。因此就有必要提供一种方法，区分开这些命令和文本。命令既可以由不能做为普通文本的单个字符组成，也可以由单词组成，它们前面都要加上一个特殊的字符，即反斜杠 \ 。

2.1.2　LaTeX 文档的结构

在每个 LaTeX 文档中，都必须有导言 (preamble) 与正文 (body) 两部分。

导言是一组命令的集合，它指定处理后面文本的全局参数，如页面格式，文本的高度和宽度，输出页的页码、页眉与页脚的组成。即使是在最简单的情形中，导言也必须包含命令 `\documentclass`, 以指定文档的全局处理类型。这通常也是导言中的第一条命令。

如果在导言中再没有其他命令，LaTeX 就会为行宽、页边、段落间隔、页面高度与宽度以及其他度量选择默认值。在以前的 LaTeX 版本中，这些设置采取的是美国标准。对于欧洲用户所用的程序，存在内建的选项，使文本的高度和宽度符合 A4 纸标准。现在已经有了与语言相关的宏包，可以把 'Chapter' 和 'Abstract' 的标题翻译成相应语言。而将在第十章中讲述的 CCT emTeX 则是通过对类文件进行修改而做到这一点的。

导言用 \begin{document} 表示结束。在第 8 页给出的 LaTeX 文件示例模板中，\begin{document} 命令前面的部分，就是一个典型的导言结构。在 \begin{document} 命令后面的所有内容都被看成正文。它由文本混杂命令组成。与导言的内容相比，这些命令只有局部的作用，即它们只适用于一定的范围，如缩进、公式、对字体的暂时改变，等等。正文是用 \end{document} 结束的。这通常也是文件的结束。

一个 LaTeX 文件的通常语法如下：

```
\documentclass[ 选项 ]{ 类 }
 其他全局命令和定义
\begin{document}
 文本与只有局部作用的命令的混合
\end{document}
```

我们将在 3.1.1 节中介绍所有可以出现在 \documentclass 命令中的选项和类。初学者可以照抄我们在第 8 页上给出的示例。随着学习进程的推进，很快就会明白所有命令的功能。

如果所用的是 LaTeX2.09，或者与之兼容，但在 LaTeX 2_ε 之前出现的版本，就必须用 \documentsytle 命令取代 \documentclass 命令。LaTeX2.09 的一般性语法应该是：

```
\documentstyle[ 选项 ]{ 类 }
. . . . . .
\begin{document}
. . . . . .
\end{document}
```

2.1.3 LaTeX 的处理模式

在处理过程中，LaTeX 总是处于下面三种模式之一：

1. 段落模式，
2. 数学模式，
3. 从左到右 (LR) 模式。

段落模式也就是正常的处理模式，这时 LaTeX 把输入文本做为一串要被 (自动) 断成行、段落和页的单词和句子。

当遇到特定命令，表示下面的文本代表公式时，LaTeX 就会切换进入数学模式，在公式中空格被忽略。例如，ln 和 l n 都解释成 l 与 n 的乘积 ln。当遇到相应的表示公式结束的命令时，LaTeX 就会切换回段落模式。

LR 模式类似于段落模式，这时 LaTeX 从左到右处理输入文本，把它们做为一组不能被断行的单词来看待。例如，当普通文本被嵌入到公式中时，或者用命令 \mbox{短文本} 来强迫 短文本 位于同一行时， LaTeX 就处于 LR 模式。

理解与识别处理模式是相当重要的，因为有些命令只能用在特定的模式中，或者在不同的模式中有不同的作用。今后我们将把段落模式和 LR 模式合称为文本模式，以表明它们的性质是一致的。但我们还是要把它们与数学模式区分开，因为这两者之间的差别是很大的。

2.1.4 生成 LaTeX 文档

从输入文本到在打印机上得到的 LaTeX 排版结果，是由三步组成的。首先利用计算机的编辑器创建 (或修改) 文本文件。这个文本文件由实际的文本混杂 LaTeX 命令组成。文本文件的全名由基本名加上扩展名 .tex 组成 (如 sample.tex) 。如果用的是 CCT 中文 LaTeX，那么后缀就应当为 .ctx，但我们要用另外的程序把它翻译成 .tex 文件。计算机操作系统通常对文件名的选取有一些限制，如基本名和扩展名可以拥有的最多字符数，或者哪些字母不可以使用，等等。例如，如果基本名只限于不超过 8 个字符，那么就可以使用 finances.tex 做文件名，而 financial.tex 是不可以的。

然后用 LaTeX 处理文本文件。在 1.1.1 节中已做了介绍，这时用 LaTeX 格式执行 TeX 程序。在大多数安装版本中，对此都有一个特定的缩略命令，即 latex2e，只要在该命令后输入文本文件的名称就可以了，不必带后缀 .tex 。例如，如果文本文件的名称是 sample.tex，那么应该用下面的调用来启动 LaTeX 处理程序:

```
latex2e sample
```

在这一处理过程中，终端监视器会显示页号以及可能会有的错误和警告消息。附录 D 介绍了这些错误消息及其后果。当 LaTeX 结束了这一处理过程后，它会生成一个新的文件，其基本名不变，后缀为 .dvi 。在上面这个例子中，得到的新文件就是 sample.dvi 。

新生成的 .dvi(Device-Independent 的缩写，表示与设备无关) 文件由格式化后的文本以及与所需要的字体有关的信息组成，但是它与所用的打印机的指令无关。这种与设备无关的文件也叫做元文件 (metafile) 。

2.1.5 浏览与打印结果

为了查看处理结果，可以利用系统提供的屏幕浏览程序。目前 DOS 下的 LaTeX 系统中浏览程序一般名为 view.bat 。而目前比较好的浏览程序是中国科学院计算数学与科学工程计算所的张林波提供的 cctwin16 和 cctwin32 程序。笔者在实际中用的就是 EditPlus 结合这个浏览程序进行 LaTeX 文件处理的。该程序可以查看中文 LaTeX 生成的结果。如果正确地对 EditPlus 进行了相关配置 (见 1.4.1 节)，那么在编译完文档文件 (按 Ctrl+1) 后，只要按 Ctrl+2 就可以启动 cctwin 程序，查看 .dvi 文件。而且如果接着在修改后重新编译了文档文件，只要在 Windows 中切换 (按 Alt+Tab) 到 cctwin 程序中，其中所显示的结果会自动刷新。

最后，在 .dvi 元文件中的信息要被转化成可以在指定打印机上输出的形式，这一过