李勇编

TEX、AMS-TEX和LATEX使用简介



高等教育出版社

TEX、AMS-TEX和LATEX使用简介 李勇编 高等教育出版社

科技排版软件

TEX、AMS-TEX 和 LATEX 使用简介

李勇编

高等教育出版社

图书在版编目(CIP)数据

TEX、AMS—TEX 和 LATEX 使用简介/李勇编．—北京：高等教育出版社．2000
ISBN 7—04—009235—2

Ⅰ．T… Ⅱ．李… Ⅲ．排版—应用软件，TEX、AMS—TEX、LATEX Ⅳ．TS803.23

中国版本图书馆 CIP 数据核字(2000)第 79664 号

TEX、AMS－TEX 和 LATEX 使用简介
李勇 编

出版发行	高等教育出版社		
社　　址	北京市东城区沙滩后街 55 号	邮政编码	100009
电　　话	010－64054588	传　　真	010－64014048
网　　址	http://www.hep.edu.cn http://www.hep.com.cn		
经　　销	新华书店北京发行所		
印　　刷	北京二二〇七工厂		
开　　本	787×1092　1/16	版　　次	2000 年 12 月第 1 版
印　　张	14.5	印　　次	2000 年 12 月第 1 次印刷
字　　数	350 000	定　　价	17.10 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

版权所有　侵权必究

序

TEX 软件是美国 Stanford 大学著名计算机专家 Donald E. Knuth 无偿奉献给科学界的一套科技排版系统。十五年来，这套软件迅速流行，现在已经是国际上最为通用的科技排版软件。特别是在数学界，无论是论文的个人交流或网上交流，还是书刊的编辑排版，几乎全部使用这一软件。

众所周知，在各学科论著的排版中以数学为最难，它不仅涉及大量的数学符号和公式，而且有多层结构。TEX 既然能够完成数学论著的排版，当然对于其它科学论著的排版就更显轻而易举。不久前出现的 MsTEX (即 Music TEX) 就是在 TEX 基础上扩充出来的、专门为音乐家写五线谱用的一种 TEX 软件包。

自 1982 年以来，这套软件系统又经历了诸多发展。首先是 1982 年由美国数学会开发的 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX，它适用于撰写数学文章。其次是 1986 年由 Leslie Lamport 开发的 LATEX，它用于编排科技书籍和杂志。这期间还开发了若干其它的软件包，但名声稍逊。TEX 系统发展的重要一步是 1990 年由美国数学会及几所大学联合作出的工作，即增加了几百种数学符号，这使得 TEX 的符号系统极为完备。另一方面，L. Lamport，F. Mittelbach，C. Rowley 和 R. Schöpf 于 1989 年设立了 LATEX 第三版的研究项目，力图统一包括 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 和旧 LATEX 等多种格式而建立起一个更为完全的系统。此项目原定于 1992 年结束，但时至今日尚未完成，由此可见此项工作的繁难和严肃性。作为过渡，他们推出了 LATEX 2_ε 版，这已经在国际上流行了若干年，而且已经汉化。以上这些软件，均可在网上下载。

在国际上，还流行着几种符号运算软件 (如 Mathematica 和 Maple 等)。这些用于科学计算的软件和 TEX 排版系统的教学乃是目前在我国高校开设的“数学实验”新课程的基本内容，急需教学参考书。鉴于普及这些软件的紧迫性，本届的教育部高等学校数学与力学教学指导委员会成立了专门的“数学软件工作小组”，负责统筹指导这项工作。本书就是根据该小组的提议编写的。作者李勇博士，是该小组的五名成员之一。他使用 TEX 已有十年，经验丰富。而且本书的前三分之二部分，曾作为讲义使用多年，较为成熟。

无疑地，本书将成为广大科教、出版界工作者的必备工具书。值此书出版之际，略述数语，以表达本人对李勇博士辛勤劳动的敬意。乘此机会，感谢教育部高教司、国家自然科学基金委员会天元基金对数学软件小组的资助，感谢高等教育出版社对出版本书的鼎力支持。

陈木法

2000 年 5 月 7 日

引 言

1990 年初我正在北京师范大学数学系攻读博士学位，导师严士健教授与杨重骏教授和汪嘉冈教授为了向国外同行介绍国内概率统计界的研究成果，编写《中国的概率论及其应用》一书，需要用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 排版。为完成此任务，陈木法教授写了一个 12 页的介绍 TEX 的手稿，这成为我开始学习 TEX 软件的启蒙材料。经过近一年的努力，我们用 XT 型计算机完成了该书的排版任务。至此我深切地感受到 TEX 软件的强大排版功能，认为该软件是科技论文、报告和书籍排版的最佳工具。因此于 1991 年 12 月和向勇同志合作编写了一本讲义《 PCTEX 使用手册》：介绍 TEX 和 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 的使用方法。该讲义经过几届研究生的使用，效果很好，很多素不相识的人纷纷来信索取，由此可见 TEX 软件的魅力。经过十年时间， TEX 软件在国内学术界已经广泛流行，但还没有一本适合读者自学 TEX 的书籍。在陈木法教授的建议下，编写了此书。

编写本书的目的有二，其一是使 TEX 初学者能够在短时间内掌握 TEX 软件的一般使用规则，其二是作为一般 TEX 使用者的参考工具手册。希望本书能够节省科技工作者、研究生、本科生学习使用 TEX 软件的时间，成为编写 TEX 源文件的好助手。由于时间仓促，恐有不少错误存于书中，恳请各位读者指正，以便修改。

陈木法教授对于本书的编写特别关心，他在百忙之中抽时间阅读了本书的全部初稿，提出了许多宝贵的修改意见，并为本书写了序，在此表示特别感谢。本人还受益于何青副教授，他为我提供了许多有关 TEX 技术细节的帮助，在这里表示衷心感谢。另外，还要感谢王建波、何洋波、杨晓霞和方维硕士对本书的校正工作，他们使本书避免了许多文字和排版方面的错误。最后要感谢教育部高等学校数学与力学教学指导委员会的“数学软件工作小组”，是该小组促使本人下决心编写此书；感谢高等教育出版社对本书出版的大力支持。

李 勇

2000 年 5 月 21 日于北京

本书的使用方法

对于初学者来说，可以首先阅读第六章，挑选一个合适的源文件模版输入到计算机中。对照模版中的中文注解，就可以快速掌握源文件的结构。还可以向模版添加文本内容，建立起自己的源文件。

对于已经熟悉源文件结构的初学者来说，可以熟悉一下本书的目录。为了读者的查阅方便，本书的目录是按 TEX 的各个功能详细编写的。通过目录表，读者可以迅速找到所需功能的具体实现方法。另外读者还要注意，目录表中以 TEX 开头的标题介绍的是 TEX 基本功能，一般它们可以用在所有格式的源文件中；以 vanilla 开头的标题介绍的是 vanilla 格式中的功能，一般它都可以用于 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 源文件里，其中绝大多数的数学符号表达方式还可以用于 LATEX 2_ε 源文件中；以 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 开头的标题一般只能用于 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 源文件中，但是其中的有关数学符号和字体的命令却可以通过适当的说明用于其他格式的 TEX 源文件中；以 LATEX 2_ε 开头的标题介绍的内容，一般只能用在 LATEX 2_ε 源文件中。

对于一般的 TEX 使用者来说，不可能精确记住所有的 TEX 命令的使用方法。在绝大多数情况下，组成 TEX 命令的字母序列恰好是该命令含义的英文单词或缩写词，因此对于懂英语的人来说，TEX 命令更容易记忆。常常会遇到要确定某一个英文字母组合是否可以构成一个 TEX 命令的问题，这时可以查阅书后的索引表，该表按字母顺序列出了书中介绍的所有 TEX 命令和解释该命令的书中页码。

索引的另一个功用是速查 TEX 命令的含义。在索引表中，每条索引记录是按如下的次序安排内容的：首先是 TEX 命令或名词，其次是该命令或名词的简单解释或它所代表的符号，在其后圆括号内标出的是该命令或名词适用的环境，最后是该命令或名词在书中详细解释所在的页码。

目 录

序 i

引 言 ii

本书的使用方法 iii

第一章 TEX 简介 1

§1.1 使用 TEX 软件的流程 1

§1.2 源文件的编辑和 TEX 的保留键 3

§1.2.1 TEX 源文件中的空格与回车 3

§1.2.2 TEX 源文件中的引号、破折号以及连字符 4

§1.2.3 TEX 源文件中的注释 5

§1.2.4 TEX 源文件中的 italic 字体和其它的常用字体 5

§1.2.5 TEX 源文件中的控制序列 7

§1.2.6 TEX 源文件中带有参量（ argument ）的控制词 8

§1.2.7 TEX 源文件中控制序列后的空格 8

§1.2.8 TEX 中的长度概念 9

§1.3 TEX 源文件的编译与错误信息 10

§1.3.1 TEX 的运行 10

§1.3.2 TEX 源文件中的错误 11

§1.3.3 TEX 源文件中的连字符和分行 13

§1.3.4 TEX 源文件中的其它种类的错误 14

§1.3.5 TEX 源文件中更难处理的错误信息 15

§1.3.5.1 源文件中表示字体的花括号 16

§1.3.5.2 TEX 的内存限制 16

§1.3.5.3 TEX 不认识源文件中表示文件结束的控制词 16

第二章 TEX 定义的格式文件 18

§2.1 TEX 中的格式文件与 plain 格式 18

§2.2 TEX 中的 vanilla 格式 18

§2.2.1 vanilla 格式的选用 19

§2.2.2 vanilla 格式与 plain 格式的差别 19

§2.3 vanilla 格式的特点 19

§2.3.1 vanilla 格式中文章标题和作者姓名的输入 19

§2.3.2 vanilla 格式中节标题和子标题的输入 20

§2.3.3 vanilla 格式中定理、命题和证明的输入 20

§2.3.4 vanilla 格式的其它特色 21
§2.4 vanilla 格式中的空隙、分行和非英文字母 22
§2.4.1 vanilla 格式中单词间空隙的控制 23
§2.4.2 vanilla 格式中分行点的控制 24
§2.4.3 vanilla 格式中重音符和特殊字符的打印 25
§2.5 vanilla 格式中的数学公式 26
§2.5.1 vanilla 格式中数学公式的表示方法 27
§2.5.1.1 vanilla 格式中正文里的数学公式 27
§2.5.1.2 vanilla 格式中单独列出的数学公式 29
§2.5.1.3 vanilla 格式中数学公式里的上标和下标 30
§2.5.1.4 vanilla 格式中数学公式里的分式和二项式系数 32
§2.5.2 vanilla 格式中常用数学符号的输入 33
§2.5.2.1 vanilla 格式中求和符号和积分符号的输入 33
§2.5.2.2 vanilla 格式中数学公式里的可变尺寸符号 35
§2.5.2.3 vanilla 格式中数学公式里特定函数和符号的表示 37
§2.5.2.4 vanilla 格式中单列公式的编号 39
§2.5.2.5 vanilla 格式中单列公式的对齐 40
§2.5.2.6 vanilla 格式中单列公式的分行 42
§2.5.2.7 vanilla 格式中数学公式里的矩阵与分段函数 43
§2.5.2.8 vanilla 格式中数学公式里的省略号 46
§2.5.2.9 vanilla 格式中数学公式里的重音符号和字体 47
§2.5.2.10 TEX 中数学公式里的小空隙 48
§2.5.3 vanilla 格式的练习答案 49
§2.6 TEX 中打印信件的 ltr 格式 51
§2.7 高级 TEX 命令 55
§2.7.1 TEX 源文件的快速编译 55
§2.7.2 TEX 源文件的分解 55
§2.7.3 TEX 源文件的自定义控制序列 56
§2.7.4 TEX 源文件的自定义带有参量的控制词 57
§2.7.5 TEX 的分行和分页 59
§2.7.6 TEX 中为插图等预留位置 59
§2.7.7 vanilla 格式中默认格式的修改命令 59
§2.7.7.1 vanilla 格式中单列数学公式编号的位置 59
§2.7.7.2 vanilla 格式中正文版面宽和高的调整 60
§2.7.7.3 vanilla 格式中正文版面位置的调整 60
§2.7.7.4 vanilla 格式中正文行距的调整 60
§2.7.7.5 vanilla 格式中的页码 61
§2.7.7.6 vanilla 格式中不规则右边界的设置 61

§2.7.7.7 vanilla 格式中稍窄于正文的段落 61
§2.7.7.8 vanilla 格式中的条目列表 61
§2.7.8 plain 格式正文版面的整体放大 62
§2.8 TEX 中的表和表格 63
§2.8.1 TEX 中用行对齐法编排表和表格 63
§2.8.2 TEX 中用固定列宽法编排表和表格 67
§2.8.3 TEX 中用可变列宽法编排表和表格 68
§2.9 TEX 中常用数学符号的表示 69
§2.9.1 小写希腊字母 69
§2.9.2 大写希腊字母 69
§2.9.3 大写花体英文字母 69
§2.9.4 各种运算符号 70
§2.9.5 各种逻辑关系符号 70
§2.9.6 各种常用符号 70
§2.9.7 各种箭头 71
§2.9.8 可变尺寸的数学符号 71
§2.9.9 各种积分号 72
§2.9.10 各种括号 72
§2.10 TEX 中的英文字体 72
§2.11 TEX 和 $\mathcal{AMS}$-TEX 及 LATEX 2ε 中的常用英文字符表 73

第三章 $\mathcal{AMS}$-TEX 格式 82
§3.1 $\mathcal{AMS}$-TEX 源文件的编译与特点 82
§3.2 $\mathcal{AMS}$-TEX 中源文件的组成部分与 amsppt 格式 83
§3.3 $\mathcal{AMS}$-TEX 中 amsppt 格式的版面定义 84
§3.4 $\mathcal{AMS}$-TEX 中 amsppt 格式的标题部分 84
§3.5 $\mathcal{AMS}$-TEX 中 amsppt 格式的正文部分 86
§3.6 $\mathcal{AMS}$-TEX 中用 amsppt 格式打印书 88
§3.7 $\mathcal{AMS}$-TEX 中用 amsppt 格式打印数学论文的例子 89
§3.7.1 amsppt 格式的源文件 89
§3.7.2 源文件的打印效果 94
§3.8 $\mathcal{AMS}$-TEX 中定义的扩充字体 99
§3.9 $\mathcal{AMS}$-TEX 中定义的数学符号 100

第四章 LATEX 简介 105
§4.1 LATEX 2ε 源文件的组织构成与编译 105
§4.1.1 LATEX 2ε 源文件的组织构成 105
§4.1.2 LATEX 2ε 源文件的编译 106
§4.2 LATEX 2ε 源文件中的版面定义部分 106

§4.2.1 LaTeX 2ε 源文件中的类和类选项 . . . 106
§4.2.2 LaTeX 2ε 源文件中宏程序包的调用 . . . 109
§4.2.3 修改 LaTeX 2ε 类规定的版面格式 . . . 109
§4.2.3.1 LaTeX 2ε 中的 plain 格式 . . . 110
§4.2.3.2 LaTeX 2ε 中的 empty 格式 . . . 110
§4.2.3.3 LaTeX 2ε 中的 headings 格式 . . . 110
§4.2.3.4 LaTeX 2ε 中的 myheadings 格式 . . . 110
§4.2.3.5 LaTeX 2ε 中的页码编号 . . . 111
§4.2.3.6 LaTeX 2ε 中与段落有关或与页面有关的控制 . . . 111
§4.2.4 LaTeX 2ε 中特定长度单位或量的赋值方法 . . . 112
§4.2.4.1 LaTeX 2ε 中特定长度单位的赋值 . . . 113
§4.2.4.2 LaTeX 2ε 中特定长度单位的修改 . . . 113
§4.2.4.3 LaTeX 2ε 中设置特定长度单位到字符串的长度 . . . 113
§4.2.4.4 LaTeX 2ε 中设置基线间距离的方法 . . . 114
§4.3 LaTeX 2ε 源文件中的正文部分 . . . 114
§4.3.1 LaTeX 2ε 源文件中的环境 . . . 114
§4.3.2 LaTeX 2ε 中的文章标题、作者和写作日期 . . . 114
§4.3.3 LaTeX 2ε 中的摘要 . . . 116
§4.3.4 LaTeX 2ε 中的部分、章、节、段标题与自动编号 . . . 116
§4.3.4.1 LaTeX 2ε 中的部分、章、节、段标题表达方法 . . . 116
§4.3.4.2 LaTeX 2ε 中章、节、段标题的自动编号原理 . . . 117
§4.3.5 LaTeX 2ε 中的目录 . . . 118
§4.3.6 LaTeX 2ε 中的附录 . . . 119
§4.3.7 LaTeX 2ε 中的参考文献 . . . 119
§4.3.8 LaTeX 2ε 中编排书的源文件结构 . . . 121
§4.4 LaTeX 2ε 中的数学公式 . . . 122
§4.4.1 LaTeX 2ε 中的数学公式环境 . . . 122
§4.4.2 LaTeX 2ε 中能够自动编号的单列数学公式环境 . . . 123
§4.4.3 LaTeX 2ε 中的多行单列数学公式环境 . . . 124
§4.4.4 LaTeX 2ε 数学公式中的数学符号 . . . 125
§4.4.4.1 LaTeX 2ε 数学公式中的 array 环境 . . . 126
§4.4.4.2 LaTeX 2ε 数学公式中的 array 环境与矩阵和行列式 . . . 128
§4.4.4.3 LaTeX 2ε 数学公式中的 array 环境与表格 . . . 129
§4.4.4.4 LaTeX 2ε 数学公式中的分段函数 . . . 133
§4.4.4.5 LaTeX 2ε 数学公式中的二项式系数 . . . 133
§4.4.4.6 LaTeX 2ε 数学公式中的多行下标 . . . 133
§4.4.4.7 LaTeX 2ε 数学公式中的积分号 . . . 134
§4.4.4.8 LaTeX 2ε 数学公式中的文本 . . . 135

§4.4.4.9 LaTeX 2ε 数学公式中的字体 . . . 135
§4.4.4.10 LaTeX 2ε 数学公式中数学符号的尺寸 . . . 136
§4.4.5 在 LaTeX 2ε 中使用 AMS-TeX 的数学符号 . . . 137
§4.5 LaTeX 2ε 中的列表 . . . 137
§4.5.1 LaTeX 2ε 中 itemize 环境列表样例 . . . 137
§4.5.2 LaTeX 2ε 中 enumerate 环境列表样例 . . . 138
§4.5.3 LaTeX 2ε 中 description 环境列表样例 . . . 138
§4.5.4 LaTeX 2ε 中嵌套列表与修改默认标签 . . . 139
§4.5.4.1 itemize 标签的修改 . . . 139
§4.5.4.2 enumerate 标签的修改 . . . 139
§4.6 LaTeX 2ε 中的自动编号与交叉引用 . . . 139
§4.7 LaTeX 2ε 中的作图 . . . 140
§4.7.1 LaTeX 2ε 中的图形环境 . . . 140
§4.7.2 LaTeX 2ε 图形环境中的坐标系 . . . 142
§4.7.3 LaTeX 2ε 图形环境中的图形设置命令 . . . 142
§4.7.4 LaTeX 2ε 图形环境中的基本绘图命令 . . . 143
§4.7.4.1 LaTeX 2ε 图形环境中的文本 . . . 143
§4.7.4.2 LaTeX 2ε 图形环境中的直线 . . . 144
§4.7.4.3 LaTeX 2ε 图形环境中的箭头 . . . 144
§4.7.4.4 LaTeX 2ε 图形环境中的矩形 . . . 145
§4.7.4.5 LaTeX 2ε 图形环境中的圆 . . . 147
§4.7.4.6 LaTeX 2ε 图形环境中的圆角矩形 . . . 147
§4.7.4.7 LaTeX 2ε 图形环境中的竖直文本 . . . 148
§4.7.4.8 LaTeX 2ε 图形环境中的有框文本 . . . 149
§4.7.4.9 LaTeX 2ε 图形环境中的 Bézier 曲线 . . . 149
§4.7.4.10 LaTeX 2ε 图形环境中线的粗细 . . . 150
§4.7.4.11 LaTeX 2ε 图形环境中嵌套图形 . . . 151
§4.7.4.12 LaTeX 2ε 中部分图形的储存与调用 . . . 151
§4.7.4.13 emtex 中的画线命令 . . . 152
§4.8 LaTeX 2ε 画图辅助软件 texcad 简介 . . . 153
§4.8.1 texcad 软件的运行与主菜单窗口 . . . 154
§4.8.2 texcad 软件的绘图命令菜单及绘图 . . . 155
§4.8.2.1 绘图命令菜单的 Text 选项与文本输入 . . . 155
§4.8.2.2 绘图命令菜单的 Framebox 选项与矩形绘制 . . . 156
§4.8.2.3 绘图命令菜单的 Line 选项与直线的绘制 . . . 156
§4.8.2.4 绘图命令菜单的 Circle 选项与空心圆的绘制 . . . 156
§4.8.2.5 绘图命令菜单的 Bezier 选项与 Bézier 曲线的绘制 . . . 156
§4.8.3 texcad 软件的图形编辑菜单与图形编辑 . . . 157

§4.8.3.1 图形编辑菜单中的 Text-Edit 选项 . . . 157
§4.8.3.2 图形编辑菜单中的 Copy 菜单 . . . 158
§4.8.3.3 图形编辑菜单中的 Save Macro 选项 . . . 159
§4.8.3.4 图形编辑菜单中的 Load Macro 选项 . . . 159
§4.8.4 texcad 软件的屏幕显示比例缩放菜单与图形屏幕显示缩放 . . . 160
§4.8.4.1 屏幕显示比例缩放菜单中的 Factor 选项 . . . 160
§4.8.4.2 屏幕显示比例缩放菜单中的 Area 选项 . . . 160
§4.8.5 texcad 软件的绘图窗口内容平移菜单与窗口内容平移 . . . 161
§4.8.6 texcad 软件的图形存盘 . . . 161
§4.8.7 texcad 软件的载入图形 . . . 161
§4.8.8 texcad 软件的退出 . . . 161
§4.9 LaTeX 2ε 中的插图 . . . 162
§4.9.1 在 LaTeX 2ε 源文件中插入 PostScrip 图形的方法 . . . 162
§4.9.1.1 在 LaTeX 2ε 源文件调用图形文件包 . . . 162
§4.9.1.2 在 LaTeX 2ε 源文件中插入剪裁图形 . . . 162
§4.9.1.3 在 LaTeX 2ε 中把插图缩放成给定的尺寸 . . . 163
§4.9.1.4 在 LaTeX 2ε 中把插图按比例缩放 . . . 164
§4.9.1.5 LaTeX 2ε 中插图的水平反射 . . . 165
§4.9.1.6 LaTeX 2ε 中插图的旋转 . . . 166
§4.9.2 把 .dvi 文件转换成 ps 格式文件的方法 . . . 166
§4.10 LaTeX 2ε 中的长表格 . . . 168
§4.11 LaTeX 2ε 中源文件的分解 . . . 170
§4.12 LaTeX 2ε 中索引的建立 . . . 172
§4.12.1 LaTeX 2ε 源文件中的索引纪录 . . . 172
§4.12.2 LaTeX 2ε 源文件中 .idx 文件与索引 . . . 173
§4.12.3 makeindex 工具与 LaTeX 2ε 的自动索引 . . . 174

第五章 中文 TeX 176

§5.1 CCT 系统里中文 TeX 源文件的建立与编译流程 . . . 176
§5.1.1 CCT 中的汉字字体与字号 . . . 176
§5.1.2 CCT 源文件的编译 . . . 177
§5.1.2.1 plain 、 vanilla 和 ltr 格式 .ctx 文件的编译 . . . 177
§5.1.2.2 AMS-TeX 格式 .ctx 文件的编译 . . . 177
§5.1.2.3 LaTeX 2ε 格式 .ctx 文件的编译 . . . 178
§5.2 在 window9x 环境中显示 .dvi 文件 . . . 178
§5.3 在 window9x 环境中编写和编译 TeX 源文件 . . . 178
§5.4 天元系统简介 . . . 181
§5.4.1 TYWIN 的特点 . . . 181

§5.4.2 TYWIN 的安装与卸载 182
§5.4.3 TYWIN 的配置 182
§5.4.4 天元源文件的编写 183
§5.4.5 在 TEX 中用 TeXDraw 绘图 184

第六章 $\mathcal{AMS}$-TEX 和 LATEX 快速入门 185
§6.1 $\mathcal{AMS}$-TEX 快速入门 185
§6.1.1 英文 amsppt 格式模板 185
§6.1.2 中文 amsppt 格式模板 188
§6.2 LATEX 2_ε 快速入门 191
§6.2.1 英文 LATEX 2_ε 模板 191
§6.2.2 中文 LATEX 2_ε 模板 191

第七章 TEX 软件包的相关网站及最新发展简介 193
§7.1 与 TEX 有关的一些网站 193
§7.2 运行于 Windows 环境的 TEX 软件包：MikTeX 193
§7.2.1 MikTeX 软件包的下载与安装 194
§7.2.2 MikTeX 的几个有趣的特点 194
§7.2.2.1 在编译源文件的同时产生定位信息 194
§7.2.2.2 建立源文件编辑器与 MikTeX 之间的联系 194
§7.2.2.3 建立 Yap.exe 与源文件编辑器间的联系 195
§7.2.2.4 Yap.exe 与源文件编辑器之间的切换 196
§7.2.3 TEX 软件包 Omega 的特点 197
§7.3 Windows 环境中的 TEX 源文件编辑器：WinEdt 197
§7.3.1 WinEdt 的安装 197
§7.3.2 WinEdt 的功能简介 198

参考文献 199

索 引 200

第一章　TEX 简介

TEX 是 D. E. Knuth 研制开发的一种科学论著排版软件。它不仅可以使你用计算机来完成一般文章的编排工作，还可以编排高质量的科学论著。TEX 的功能是强大的，本书就是利用该软件编排的，其中的目录、章节页码的引用、公式编号的引用以及参考文献编号的引用都是由 TEX 自动生成的，不需要人为干预。TEX 排版速度极快，在 PentiumIII（500MHz）计算机中，本书全文的编译时间不到 6 秒钟。它是目前国际上公认的最好的科学论著排版软件。用 TEX 可以打印各种数学式子，如：

$$\sum_{l:\ \text{奇数}} \binom{n}{l} = 2^{n-1} \quad \text{和} \quad \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

TEX 的原型为希腊文的 $\tau\varepsilon\chi$，是希腊文“艺术”一词的头三个字母，也是英文单词 technology 的词根。因此，Knuth 建议用 TEX 作为这种能够打印高质量科技文章软件的名称。

TEX 由一些最基本的命令组成，它们可以完成一些简单的排版操作和程序设计功能。美国数学会用 TEX 宏命令设计的 $\mathcal{AMS}$-TEX 软件包使得 TEX 能方便地编排出绝大多数数学符号以及数学论文格式；而 LATEX 软件则使 TEX 能够完成打印高质量书籍、报告和文章的任务。

目前有多种运行环境版本的 TEX，如 Unix 环境的、Dos 环境的和 Windows 环境版本的 TEX。在 Dos 环境下运行的 TEX 软件包中，较为著名的是 Emtex 和 Pctex；在 Windows 环境下运行的 TEX 软件包中，较为著名的是 Pctex32 和 Scientific WorkPlace 软件包。其中 Scientific WorkPlace 软件包是所见即所得的 TEX 软件包，并且它还集成了著名符号运算软件包 Maple 的一些功能，使得在编排数学文章的同时可以对其中的公式进行符号或数值运算，成为一个科学工作平台。遗憾的是上面的这些软件都是英文软件包，不能直接用于带有汉字文章的排版。要处理汉字文章的排版，需要由特别编写的转换软件。在这类转换软件中，比较著名的是 Cct 系统和天元（Ty）系统，它们可以帮助我们用 TEX 处理汉字。

对于 TEX 软件包来说，无论它是运行于何种环境中，它的源文件的格式都是一样的。所谓 TEX 源文件就是由 TEX 命令和普通文章的语句构成的文件。TEX 对源文件进行编译可以生成供 TEX 驱动程序使用的文件，通过这些文件就可以进行屏幕显示和打印输出。

在本书中，我们以运行于 TEX 环境中 Emtex 软件包（3.14159[4a] 版）为例介绍 TEX、$\mathcal{AMS}$-TEX和 LATEX的使用方法。本章中我们简单介绍 TEX 的一些最基本的知识。

§1.1　使用 TEX 软件的流程

TEX 的使用很方便，首先要进入执行程序 tex.exe 所在的子目录 **\emtex** 中。在这里你可以找到 TEX 软件包的主程序 tex.exe，它是用来完成对（用 TEX 语言编写的）源文件进行编译的程序。

下面说明 TEX 的使用流程，先编写一个简单的 TEX 源文件 test.tex 。这个源文件应该是以纯文本的方式编写的，只包括如下的一段内容：

```
This is a very short test, to make sure the program is working
correctly.  This paragraph starts  flush left, and shows the
appearance of {\bf bold face} type.

This paragraph is indented, and shows the appearance of {\it italics}.
It contains the math formula $z^n=x^n+y^n$. Such a formula might also be
displayed
$$z^n=x^n+y^n$$
to make it more prominent.

\bye
```

现在可以对这个源文件用 TEX 进行编译。为此，键入

```
tex test.tex ↩
```

这里和以后我们总是用 "↩" 表示回车键。当缺省文件的扩展名时， TEX 总是认为文件的扩展名为 .tex ，所以可键入

```
tex test ↩
```

来代替前面的命令。

如果 TEX 安装正确，则在屏幕上可以得到信息说 test.dvi 文件和 test.log 文件已被建立。其中 test.log 文件是普通的文本文件，它记录了编译源文件 test.tex 的过程中所产生的排版信息(包括错误信息)，可以用 DOS 的 type 命令或者通过文字处理软件把这个文件在屏幕上显示出来；而 test.dvi 文件不是普通的文本文件，不能用 type 命令在屏幕上显示出来，它是供打印驱动程序使用的文件。现在你可以在屏幕上显示排版结果了，为此只需键入

```
view test.tex ↩
```

屏幕上将会出现如下的内容：

This is a very short test, to make sure the program is working correctly. This paragraph starts flush left, and shows the appearance of **bold face** type.

This paragraph is indented, and shows the appearance of *italics*. It contains the math formula $z^n = x^n + y^n$. Such a formula might also be displayed

$$z^n = x^n + y^n$$

to make it more prominent.

如果你想用打印机输出，则需要根据不同的打印机型号选择相应的 TEX 打印机驱动程序. 假设我们的计算机连接的是 24 针点阵打印机，可以选用打印驱动程序 dvi24p.exe (它是 CCT 程序包中的一个打印机驱动程序，通常可以用它来驱动 24 针点阵打印机或 Canon 喷墨打印机，其详细使用说明见参考文献 [11]) 来打印排版结果. 在联通打印机后键入

```
dvi24p test ↩
```

即可在打印机上输出排版的结果. 总之，使用 TEX 编排打印科技文章 (书籍) 的步骤如下：

- 用文件编辑器建立好源文件（方法见下一章）.
- 用 TEX 编译排版、查错。
- 用 TEX 提供的驱动程序进行屏幕显示或打印输出最后排版结果。

对于各种 TEX 软件包（包括 $\mathcal{AMS}$-TEX 和 LATEX 等），均使用上述步骤来编排文章或书籍。

§1.2 源文件的编辑和 TEX 的保留键

在这一节里，我们将通过例子来介绍用 TEX 编辑源文件的一般准则以及一些特殊的字符(如斜体字、符号 †和 ‡等) 的输入方法。

编辑 TEX 源文件需要一个文件编辑器。一般来说任何文件编辑器都可以用来编写 TEX 源文件，如 Windows 环境的 editplus 、 Word ， Dos 环境的 Qe 和 Editor ， wps 等。要注意的是：当使用类似于 Word 这样的编辑器时，一定要选择纯文本（有些软件也叫做非文书文件）格式来编写 TEX 源文件。

在 TEX中使用了 12 个保留键，它们在 TEX 中有特定的含义，需要特别留心。这些保留键分别是：

\ $ { } % ~ @ # ^ _ " &

这些键的应用构成了 TEX 语言的主要特征，它们的用法将在后面详细介绍。此处只简单说明其中几个符号的用法。

1. 反斜扛 “ \ ” 的后面紧跟字符产生控制序列，表示特定的 TEX 命令。例如 “ \{ ” 在 TEX 中表示左花括号 “ {”。
2. 花括号 “ { ” 和 “ } ” 表示控制序列的作用区间或源文件中的子体结构。例如在下面一段

   ```
   \underbar{This is a} word
   ```

   输入中它表示在花括号内的词组下面划线，打印出即为：

   This is a word

   在这种情况下，左、右花括号必须成对出现，否则会出现较大的错误。这是因为 TEX 一旦发现了一个左花括号 “ { ”，它就要寻找与之配对的右花括号 “ } ” 以完成一个子体结构。
3. 百分号 “ % ” 用于源文件中表示它后面的同一行的所有字符均被认为是注释行而不被 TEX 打印出来。

除此之外， “ + = < > ” 这四个符号只能用在数学公式中，否则在 TEX 的打印输出中会产生奇怪的符号。

§1.2.1 TEX 源文件中的空格与回车

首先，让我们用文件编辑器建立一个名为 try.tex 的源文件，以便能够试验 TEX 的各种性能。

TEX 把源文件中两个单词之间的所有空格都看作一个空格；把按一次回车看作一个空格，而把连续按两次回车 (即在屏幕上出现一个空行) 看作一个新自然段的确开始。在我们的源文件 try.tex 中键入如下的内容：

```
This is   the first paragraph. Nothing
```

```
fancy here, and it dosn't look
 too    good.
  but it will   look a  lot
better   when it's printed. To
    see why, just keep reading!
```

在上面的例子中，各行的长短不同，并且有的单词之间空了几个格，这无关紧要。因为源文件中每一行的长短并不影响 TEX 的排版。同样源文件中单词之间的空格数量也不影响 TEX 的排版。当你编辑源文件时，可以常按回车键，使得每一行输入的字符不超过屏幕所能显示的范围，以提高效率。上面这段源文件所对应的打印输出为：

This is the first paragraph. Nothing fancy here, and it dosn't look too good.
but it will look a lot better when it's printed. To see why, just keep reading!

TEX 规定一个段落的结束用一个空行来表示。因此当你要另起一段时，仅需要连按两次回车键，就可以开始新一段的输入。现在可以在源文件 try.tex 结尾处连续输入两个回车后，键入下面的一段内容：

```
This is the second paragraph. First we'll
print the number    186,384.64.
(That's the
speed of light in miles/second.
I measured it in myself on 3/1/85).
```

注意数字“186,384.64.”中的逗号和第一个句号的后面没有空格。但是第二个句号是一个句子的结尾，所以在它的后面至少有一个空格或回车键。现在源文件 try.tex 对应的打印输出如下：

This is the first paragraph. Nothing fancy here, and it dosn't look too good.
but it will look a lot better when it's printed. To see why, just keep reading!
This is the second paragraph. First we'll print the number 186,384.64.
(That's the speed of light in miles/second. I measured it in myself on 3/1/85).

§1.2.2 TEX 源文件中的引号、破折号以及连字符

现在让我们在 try.tex 中添加第三段内容 (注要在第二段的结尾处连按两次回车键以表示第三段的开始)：

```
Hyphens are needed whenever we want to get
`compound words, like
``mother-in-law ´´´.  Both upper-
and lower-case letters can occur,as in the word ``X-rated´´.
```

你可能会猜到 TEX 会把上面一段文字中的“`”和“´”分别打印成左单引号和右单引号，而把“``”和“´´”分别打印成左双引号和右双引号。要注意的是右双引号在源文件中是用连续输入两个键盘上的右单引号键“´´”来表示，而不是键盘上的“"”键。

注意前面例子中 upper- 的后面按了一次回车键，这使 TEX 的输出会在 upper- 的后面产生一个空格，此例中这正是我们想让 TEX 打印出来的。通常在源文件中连字符的后面不应该有空格或回车，否则 TEX 在打印输出时就会给出一个相应的空格。例如，当你想打印复合词

mother-in-law 时，在源文件中就不能把该词分成两行来输入。

在源文件中连续键入两个连字符相当于一个破折号，而连续输入三个连字符相当于一个更长一点的破折号。

§1.2.3 TEX 源文件中的注释

有时你会想在源文件中加某些注释，同时又不想让 TEX 把这些注释打印出来。这时你可以使用键盘上的 % 键。当 TEX 看见百分号 % 后，就把百分号和在它之后的同一行内容看作注释而不把它们打印出来。例如，当源文件中有下面的一段

```
I learned of this idea from
John Doe. % Remember to visit
% him in the hospital
who thought of it while sky-diving.
```

时，则这一段的 TEX 打印输出为：

I learned of this idea from John Doe. who thought of it while sky-diving.

要特别强调的是源文件中的 “ Doe. ” 和符号 % 之间的空格不能缺少，否则 TEX 不会在这个英文句号的后面留出空格，从而上面的这段打印输出变为：

I learned of this idea from John Doe.who thought of it while sky-diving.

出现这样结果是因为百分号使得 TEX 不理睬它之后同一行的回车键的输入，所以在源文件中可以利用它把一个英文单词分两行来输入。例如，当你输入

```
who suddenly thought of it while sky-%
diving.
```

后， TEX的输出为：

who suddenly thought of it while sky-diving.

即 TEX 不在连字符后面打印空格。

§1.2.4 TEX 源文件中的 italic 字体和其它的常用字体

当 TEX 开始对源文件进行排版时，它必须知道应该使用什么样的字体。当你使用缺省格式 (plain style) 时， TEX 将自动选用 10 point 的 computer mordern roman 字体。如果你需要把某些字符用其它的字体打印时，就必须使用一些特殊的 TEX 能够执行的命令。

在源文件中出现的英文字母都是你想打印出来的字母，所以必须有一个特定的设置来表示 TEX 命令。为此， TEX 选用了反斜杠键 “ \ ” （普通的手动打字机上没有这一键）来表示这种设置。当 TEX 在源文件中读到反斜杠 \ 时，它就知道紧跟其后的是一个要执行的命令，而不是一个字符。

大多数的 TEX 命令都是由反斜杠 \ 和紧跟其后的字符序列构成的，并且这些字符序列通常组成一个英文单词或缩写词，其含义通常和该 TEX 命令的含义有密切的联系，很容易记忆。例如 \it 是让 TEX 开始选用 italic 字体（即斜体字），其中的 “ it ” 恰好是英文单词 “ italic ” 的头两个字母。现在可以在源文件中输入

```
to swich to \it italic type.
```

注意，在命令 \it 的后面至少要有一个空格或紧跟在它后面的不是英文字母。如果忘记了这一点，输入的是

```
to swich to \ititalic type.
```

TEX 就会认为你使用了命令 \ititalic，而在 TEX 中没有规定这一命令的定义，从而在排版时将会产生错误信息 (我们将在下一节介绍错误信息)。另外，TEX 还区分命令中的大写和小写字母。例如，当你想选用 italic 字体的时候，就不能使用 \IT 或 \It。

\rm 也是 TEX 命令，它用来指示 TEX 选用 roman 字体。如果输入

```
This is \it italic type. \rm This is roman again.
```

则可以得到如下的打印输出：

This is *italic type.* This is roman again.

通常，我们不选用前面的方法来输入上面的文字，而是用下面的更简单的方法：

```
This is {\it italic type.}  This is roman again.
```

像反斜杠键一样，键盘上的花括号键“{”和“}”也是 TEX 的保留键。它表示花括号之间的内容在整个源文件中是一个子体。当 TEX 读到 {（表示子体的开始）时，它就记住前面正在使用的字体，而当 TEX 读到和这个左花括号对应的右花括号 }（表示子体的结束）时，它就恢复子体之前所使用的字体。因此，在上面的例子中，右花括号 } 恢复了 roman 字体。

有时你会发现 italic 字体和 roman 字体交界处的空太小，例如

not terrific

中的 *not* 和 terrific 之间的空隙就太小。为解决这一问题，需要在这两种字体的交界处再加一点额外的空。TEX 命令“\/”就是用来完成这一任务的。所以

```
{\it not\/}  terrific
```

的打印结果为：

not terrific

为了得到漂亮的打印输出，总是应该在 italic 字体结束时使用斜体调整符 \/，除非后面的字符是标点符号。

现在回到我们的源文件 try.tex 上来。让我们在它里面添加如下一段内容以复习刚刚学过的命令：

```
You have to be {\it care\/}ful and {\it
thought\/}ful with {\it slanted\/}  fonts.
You often need the ``{\it italic correc%
tion\/}'' for proper spacing, though not
before a {\it comma}, or a {\it period}.
```

TEX 中还有 *slanted* 字体，它可以通过命令 \sl 来实现该字体的选择。\sl 命令的使用方法和 \it 相同。现在可以在 try.tex 中添加另外一段，新一段的内容和上一段的唯一不同之处是 \it 改成 \sl：

```
You have to be {\sl care\/}ful and {\sl
thought\/}ful with {\sl slanted\/}  fonts.
You often need the ``{\sl italic correc%
```

```
tion\/}'' for proper spacing, though not
before a {\sl comma}, or a {\sl period}.
```

在 TEX 中，命令 \bf 用来表示选用黑体字。让我们在 try.tex 中输入如下的一段：

```
When we {\bf realy}  want to come on
strong. we can start \bf shouting in
bold face type. If we have a lot to
say, we might want to dispense with
``grouping'' and just remember to get
back to ordinary type when we're worn out.\rm
```

§1.2.5 TEX 源文件中的控制序列

上一段中介绍的命令 \it 和 \sl 都是控制序列。控制序列是由反斜杠和紧跟其后的字符序列所构成。我们把控制序列分成两大类：一类称为控制词，它是由反斜杠和紧跟其后的字母序列所构成；另一类叫做控制符，它是由反斜杠和紧跟其后的一个非字母符号所构成。\it 、\bf 和 \rm 都是控制词，而 \/ 为控制符。下面再介绍几个常用的控制序列。

在 TEX 中，控制词 \par 表示一个自然段的结束。但是在前面我们并没有提到过它，是因为它与一个空行的作用相同。另外，TEX 还把两个以上的空行看作一个空行。

人们习惯用符号 § 作为文章中节的记号，在 TEX 中这一符号用控制词 \S 来表示。我们可以在源文件中输入：

```
which we learned about in the previous \S,
```

我们曾在前面强调过控制词 \it 的后面必需至少要有一个空格或紧跟其后的不是文字，大家自然要问为什么这里控制词 \S 的后面紧跟的是逗号而不是空格？因为 TEX 规定控制词是由反斜杠后面紧跟字母构成的，所以 " \S, " 不是控制序列，从而 TEX 读到逗号时就知道从反斜杠到逗号之前的部分为控制词了。问题是当控制词 \S 与逗号之间有空格时，即输入的是

```
which we learned about in the previous \S ,
```

时 TEX 会有什么样的打印输出结果？你可能会担心 TEX 在打印输出时会在 § 和逗号之间放一个空格。不必这样担心，因为 TEX 总是不把控制词后面的空格打印出来。

符号 ¶ 、† 、‡ 和 © 都是一些特殊的符号，在 TEX 中这些符号都可以用控制词来表示：

\P	\dag	\ddag	\copyright
¶	†	‡	©

现在可以在我们的 try.tex 中再添加新的一段内容：

```
There are special ``control sequence''
to produce the symbols \S, \P, \dag
, \ddag, and \copyright .
```

我们不但对于像 § 、¶ 、† 、‡ 和 © 这样的键盘上没有的符号需要使用控制词，而且对于 TEX 的保留键上的符号也需要用控制词来表示。例如，键盘上的符号键 % 是 TEX 的保留键，因

此 TEX 用控制符 \% 来表示百分号 % 。还有一些类似的控制符表示保留键上的符号：

```
\%  \$  \&  \#  \_
```

% $ & # _

§1.2.6 TEX 源文件中带有参量（ argument ）的控制词

除了用控制序列产生一些特殊的符号之外，还可以用它来把某些词组的下面划线．例如，当我们输入

```
\underbar   {underline} a word.
```

时， TEX 的相应的打印输出为：

underline a word.

\underbar 是 TEX 的另外一种控制词，它告诉 TEX 在打印输出时以某种特殊的方式处理某些字符．我们称这类控制词为带有参量的控制词．这里的参量指的是那些要特殊处理的字符，也就是跟在控制词后的花括号之间的那些字符．如果跟在带参量的控制词的第一个字符 (不包括空格和回车键) 不是花括号，则 TEX 仅把这一字符按规定的方式处理．当你输入的是

```
\underbar underline a word.
```

时，相应的打印输出为：

underline a word.

注意观察下面两个输入

```
{\it underline\/} a word.
```

和

```
\underbar {underline} a word.
```

之间的差别．第一个输入中的花括号之间的字符为源文件中的子体，这个子体为控制词 \it 的作用范围；而第二个输入中花括号之间的字符则是要带参量的控制词 \underbar 特殊处理的参量。注意不要把它们的用法弄混．如果你输入的是：

```
\it {italicize}   a word.
```

则 TEX 在相应的打印输出时会把控制词 \it 后的所有字母（包括花括号后面的字母）都用 italic 字体打印出，即相应的打印输出结果为：

italicize a word.

§1.2.7 TEX 源文件中控制序列后的空格

我们已经知道 % 、 $ 、 & 和 _ 都是控制符，另外还有一个控制符是由反斜杠跟一个空格所构成，即 \␣（当特别需要强调时我们用 ␣ 表示空格），它是让 TEX 打印一个空格．因为 TEX 不把控制词后的空格打印出来，所以当你在源文件中输入

```
the previous \S of this Lesson.
```

后，你能得到的相应的打印输出为：

the previous §of this Lesson.

为使符号 § 和单词 of 之间有一个空格，可以使用控制符 \ ，即应该输入

```
the previous \S\ of this Lesson.
```

等价地，还可以用如下的方法输入：

```
the previous {\S} of this Lesson.
```

或

```
the previous \S{} of this Lesson.
```

控制符和控制词不同。控制词可以任意长，而控制符仅由反斜杠和紧跟其后的一个非字母符号构成。因此，控制符的后面可以紧跟其它字符。例如，你可以在源文件中输入

```
He ate 3 \# size boxes of M\&M's.
```

TEX 是按如下的方式处理控制序列后的那些空格的：当它为控制词时，TEX 在打印输出时不打印其后的任何空格；当它为控制符时，TEX 在打印输出时会把其后的空格打印出一个。

为熟悉上面介绍的内容，让我们在 try.tex 中添加如下的一段：

```
I agree 100\% with the policy that spaces
after control {\it symbols\/}   should not
be ignored. I'd pay extra \$\$ for that!
```

作为本节的结束，介绍另一个控制词 \TeX，它代表 TEX。

§1.2.8 TEX 中的长度概念

TEX 中可以使用的长度单位及相应的含义如下：

cm	厘米
mm	毫米
in	英寸 (1 in=2.54 cm)
pt	点 (1 in=72.27 pt)
bp	大点 (1 in=72 bp)
pc	pica(1 pc=12 pt)
dd	didôt 点 (1157 dd=1238 pt)
cc	cicero(1 cc=12 dd)
em	大写字母 M 的宽度
ex	字母 x 的高度

TEX 中的长度有固定长度与橡皮长度之分。固定长度的长度是固定的，如 12cm 表示长度为 12 厘米的固定长度。橡皮长度是可以在一定范围内伸展或收缩的长度，例如

```
12cm plus 1cm minus 0.5cm
```

表示长度为 12 厘米的橡皮长度，该长度可以根据需要在 11.5 cm 到 13 cm 之间变化。在橡皮长度的表示中，plus 后面的长度表示橡皮长度可伸展的最大增量，minus 后面的长度表示橡皮长度可收缩的最大减量。TEX 使用橡皮长度是为了排版美观的需要。

一个有用的橡皮长度是 \fill，它的正常长度为 0，但是可以伸展到任何长度。由此衍生的橡皮长度 \hfill 表示水平方向的橡皮长度，它实际上等于 \hbox{\fill}；另一个橡皮长度 \vfill 表示垂直方向的橡皮长度，它实际上是 \vbox{\fill} 的简写。另外还有两个控制词的性质类似于橡皮长度，都具有橡皮的性质，它们是

\dotfill 和 \hrulefill

其中 \dotfill 表示可以延伸到任何长度的圆点填充， \hrulefill 表示可以伸展到任何长度的水平直线。下面一段源文件

```
\newline 橡皮长度 \hfill hfile\newline
橡皮长度 \dotfill dotfile\newline
橡皮长度 \hrulefill hfile\newline
橡皮长度 \hfill 中间 \hfill hfile\newline
左边 \dotfill 中间 \hrulefill 右边
```

所对应的打印输出为：

橡皮长度 hfile
橡皮长度 .....dotfile
橡皮长度 _____hfile
橡皮长度 中间 hfile
左边 .....中间 _____右边

§1.3 TEX 源文件的编译与错误信息

在 TEX 中控制词 \bye 表示源文件的结束。现在先让我们在上一节建立的源文件 try.tex 的结尾处输入控制词 \bye ，然后就可以用 TEX 对它进行排版和查错了。

§1.3.1 TEX 的运行

运行 TEX 的命令格式为：

```
tex [d:][path]filename↩
```

其中 d 表示源文件所在的驱动器， path 表示源文件所在的路径， filename 为源文件的全名。当源文件的扩展名为 .tex 时，可以省略扩展名。现在可以输入命令

```
tex try↩
```

则屏幕上将会出现如下的内容：

```
This is emTeX (tex386), Version 3.14159 [4a] (no format preloaded)
**try
(try.tex [1] )
Output written on try.dvi (1 page, 2032 bytes).
Transcript written on try.log.
```

这时 TEX 已根据源文件 try.ex 建立了供打印程序使用的文件 try.dvi 和记录 TEX 运行过程的文件 try.log 。有了 .dvi 文件之后，就可以在屏幕上显示或者用打印机打印 TEX 的排版结果了。在屏幕上显示排版结果的命令格式为：

```
view filename↩
```

其中 filename 为要显示的文件名（注意不要把扩展名 .dvi 输入）。用打印机打印的命令格式为（假设 TEX 的打印机驱动程序为 dvi24p.exe ）：

```
dvi24p  filename↩
```

其中 filename 为要打印的文件名（注意不包括扩展名）。

现在我们可以在屏幕上显示源文件 try.tex 的排版结果了。输入

```
view  try↩
```

之后，屏幕上就会显示出如下的内容：

This is the first paragraph. Nothing fancy here, and it dosn't look too good. but it will look a lot better when it's printed. To see why, just keep reading!

This is the second paragraph. First we'll print the number 186,384.64. (That's the speed of light in miles/second. I measured it in myself on 3/1/85).

Hyphens are needed whenever we want to get 'compound' words, like "mother-in-law". Both upper- and lower-case letters can occur,as in the word "X-rated".

You have to be *care*ful and *thought*ful with *slanted* fonts. You often need the "*italic correction*" for proper spacing, though not before a *comma*, or a *period.*

You have to be careful and *thought*ful with *slanted* fonts. You often need the "*italic correction*" for proper spacing, though not before a *comma*, or a *period.*

When we **realy** want to come on strong. we can start **shouting in bold face type. If we have a lot to say, we might want to dispense with "grouping" and just remember to get back to ordinary type when we're worn out.**

There are special "control sequence" to produce the symbols §, ¶, †, ‡, and ©.

I agree 100% with the policy that spaces after control *symbols* should not be ignored. I'd pay extra $$ for that!

§1.3.2 TEX 源文件中的错误

让我们在源文件 try.tex 中的控制词 \bye 之前添加如下的一段：

```
This was typeset by \Tex, of course.
```

因为 TEX 区分控制词中的大写和小写字母，所以上一段中的 \Tex 与 TeX 是不同的控制词。在 TEX 中没有对控制词 \Tex 定义，从而它读到这个控制词时就不知道该做什么，因此它在排版查错时就会给出信息，告诉你控制词 \Tex 没有定义。现在让我们再用 TEX 来对 try.tex 排版，看看会出现什么情况。键入

```
tex  try↩
```

后，TEX 会在屏幕上显示如下的信息后停止运行：

```
This is emTeX (tex386), Version 3.14159 [4a] (no format preloaded)
**try
(try.tex
! Undefined control sequence.
```

```
l.45 This was typeset by \Tex
                                , of course.
?
```

其中感叹号！表示在源文件中发现语法错误，同一行后面的是有关错误类型的提示；而下面一行句号后的数字为错误在源文件中的行数，跟在后面的字符是源文件中一行内容的一部分，第二行的结尾处是 TEX 发现错误的地方；第三行的开头位置是上一行的结尾位置，这一行的文字是源文件中 TEX 发现错误的那一行的位于错误之后的文字；最后一行的问号？是 TEX 问你有何命令。回答？有如下几种选择：

1. 输入

   x↩

   表示退出 TEX；
2. 输入

   r↩

   表示让 TEX 不停顿地继续运行，直到源文件的结尾；
3. 输入回车键

   ↩

   表示让 TEX 不理该错误继续运行；
4. 输入

   h↩

   表示要求 TEX 进一步解释该错误
5. 输入

   i↩

   表示你要修改这一错误，这时屏幕上会出现

   insert>

   此时就可以输入你的修改命令。注意这里的修改只会在将要生成的文件 .dvi 中起作用，而不会对源文件作任何修改。

一般人们总是喜欢选用第二项，让 TEX 不间断地继续查错，使最后产生的 .log 文件能够包含源文件的所有语法错误，然后把 .log 文件用 DOS 命令在打印机上输出或调入编辑器的另一个窗口（如果该编辑器支持多窗口的话），最后对照 .log 文件纪录的信息对源文件中的错误一一进行修改。

在前面的例子中，我们在问号的后面可以先输入

i↩

然后再在 insert> 的后面输入

\TeX↩

这表示源文件中的“ \Tex ”应该改为“ \TeX ”，从而 TEX 继续运行产生 try.dvi 文件和 try.log 文件。当你在屏幕上显示 try.dvi 文件之后会发现它的最后一行会出现“ TEX ”。

当然也可以把上面的两步合成一步，即在？的后面直接输入

```
i\TeX↩
```

以表示源文件中的 \Tex 应该改为 \TeX 。

在前面的例子中，如果在？的后面只按回车键，即输入

↩

则 TEX 也会继续运行，也会建立 try.dvi 文件和 try.log 文件。但这时你会发现屏幕上显示的 try.dvi 文件的最后一行为：

This was typeset by , of course.

缺了我们想要打印出的 'TEX' 。

§1.3.3 TEX 源文件中的连字符和分行

如何分行是任何排版系统都会遇见的问题。 TEX 可以正确确定一个大英文字典 90% 单词的可以用连字符的位置（对于常用单词正确率更高）。连字符的作用主要是使得排版的版面的右边界整齐。然而对于有些单词， TEX 就不知道它可以用连字符的位置。当这样的单词位于一行的结尾时， TEX 就不知道该如何去分行。

现在来看看 TEX 是如何处理下面这段源文件的：

```
\TeX dosn't know how to hyphenate the word
``aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa''%
---that is a word that
apparently did not make
it into \TeX's database

\TeX dosn't know how to hyphenate the word
``aaaaa\-aaaaa\-aaaaa\-aaaaa\-aaaaa\-aaaaa%
\-aaaaa\-aaaaa''%
---that is a word that
apparently did not make
it into \TeX's database
```

因为由 40 个字母 a 并不构成一个英文单词， TEX 当然不会认识它，更不会知道应该如何断这个词。上面的这段源文件对应的是两个自然段的输出，这两段没有什么大的不同，只是第二段的 40 个 a 中多了一些控制符 \- ，我们看看 TEX 相应的输出有何差别：

TEXdosn't know how to hyphenate the word "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"—that is a word that apparently did not make it into TEX's database

TEXdosn't know how to hyphenate the word "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"—that is a word that apparently did not make it into TEX's database

显然第一段的第一行超出了版面的右边界。

假设 " bbbbcccc " 是一个 TEX 不知道如何断词的单词，并且它在源文件中经常出现，可以在源文件的开头输入一行

```
\hyphenation{bbbb\-cccc}
```

以使 TEX 知道所有的单词 bbbbcccc 都可以在字母 b 和 c 之间加连字符分行。也许在源文件的开头你还想用另一行

```
\hyphenation{bbbb-cccces}
```

来告诉 TEX 单词 bbbbcccces 也可以在字母 b 和 c 之间断行。但是人们更喜欢简单地输入

```
\hyphenation{bbbb\-cccc bbbb-cccces}
```

来告诉 TEX 单词 bbbbcccc 和 bbbbcccces 都可以在字母 b 和 c 之间断行。

§1.3.4 TEX 源文件中的其它种类的错误

现在我们已经了解了一些简单的错误信息，但有时会碰到更为复杂的错误信息。让我们回到源文件 try.tex 。先把结尾的控制词 \Tex 改为 \TeX ，然后在它的后面添加下面的一段：

```
This is the \underbar{\It next\/} experiment.
```

其中的控制词 \It 在 TEX 中没有定义。当我们用 TEX 对 try.tex 进行排版时，会得到如下的一条错误信息：

```
! Undefined control sequence.
<argument> \It
                next\/
\underbar #1->$\setbox \z@ \hbox {#1
                                    }\dp \z@ \z@ \m@th \underline {\b...
l.47   This is the \underbar{\It next\/}
                                          experiment.
?
```

这一段错误信息看起来很难懂，但只要我们学会忽视其中的一部分，就容易看懂它了。

通常，一段错误信息中的倒数第三行开始显示的是错误在源文件中的行数和内容，TEX 把源文件中的有错误的那一行分成两行来显示 (即错误信息中的倒数第三行和倒数第二行)，而错误出在上面一行的结尾处 (即错误信息中倒数第三行的结尾处)。但是在上面的例子中则不是在没有定义的控制词 \It 之后分行，而是在带参量的控制词 \underbar 的参量 {It next\/} 之后分行。这是因为 TEX 首先读到的是带参量的控制词 \underbar，这时它并没有发现什么错误；接着在它读参量时才发现 \It 没有定义，结果 TEX 认为是带参量的控制词的参量出了错误。倒数第三行中的数字 47 是源文件中该错误的所在的行数。

上面这段错误信息的第二行开头的 <argument> 表示参量中发现错误，第二行后面的部分和第三行显示的是源文件中有错误的参量，而第二行的结尾正是 TEX 发现错误的地方。现在参量 \It next\/ 在控制符 \It 的后面分行，这表示 TEX 在这里发现了错误。

错误信息中很难理解的中间几行 (在这里是第四行和第五行) 是用 TEX 的语言来描述带参量的控制词的定义，这对我们理解出错信息没有什么帮助，因此我们可以不理睬这几行的内容。

总之，我们只需要分析错误信息的头几行和后几行内容，就可以知道源文件中出现了什么错误以及该错误在源文件中的位置。

我们知道，带参量的控制词 \underbar 是在它的参量下面划线。当我们输入

```
\underbar{two words}
```

时，TEX 相应的打印输出为：

two words

要说明的是 TEX 把下面划线的词组 two words 看作一个整体，从而不能把它分成两行打印。所以 \underbar 的参量不能太长，否则 TEX 就不知道如何断行，从而相应行的打印输出就会超出规定的右边界。现在让我们看看输入

```
\underbar {two

paragraphs}
```

会出现什么情况。上面的输入中，单词 "two" 和 "paragraphs" 之间有一个空行，它相当于表示一段结束的控制词 \par 。而 TEX 规定在 \underbar 的参量中不许出现控制词 \par ，所以当 TEX 读到上面一段时会给出如下的错误信息：

```
! Paragraph ended before \underbar was complete.
<to be read again>
                   \par
l.50

?
```

现在让我们回到源文件 try.tex 上来。先把最后一段中的 \It 改成 \it ，然后把表示源文件结束的控制词 \bye 去掉。现在输入

```
tex↩
```

后，虽然 TEX 不会给出任何错误信息，但是最终它会在给出一个提示符 * 后停止运行，即显示如下信息后停止运行：

```
This is emTeX (tex386), Version 3.14159 [4a] (no format preloaded)
**try
(try.tex)
*
```

记住，单个的提示符 * 表示 TEX 需要更多的源文件的输入，而提示符 ** 表示 TEX 要求输入一个具体的文件名。现在我们的源文件 try.tex 缺少表示源文件结束的控制词 \bye ，所以 TEX 读到 try.tex 的最后一个字符后不知道源文件是否该结束，从而给出提示符 * 。

当单提示符 * 出现后，你只需在它的后面输入

```
\bye↩
```

即可以表示源文件的结束。注意，这时如果在 * 的后面输入

```
i\bye↩
```

则 TEX 会认为你想先在源文件 try.tex 的后面添加字母 i ，然后再添加控制词 \bye 。

§1.3.5 TEX 源文件中更难处理的错误信息

有时我们会遇到使人不解的错误信息，这时我们可以查阅下面的几段内容，以帮助找到源文件中的问题所在。

§1.3.5.1 源文件中表示字体的花括号

即使源文件结尾有表示结束的控制词 \bye，在源文件中也可能出现这样的错误：直到 TEX 读到表示源文件结束的控制词 \bye 时它才会有错误。假设源文件中有如下的内容：

```
Let's start {\it italic type here ...
```

并且你忘了输入表示子体结束的右花括号 }。则当 TEX 对这个源文件排版时，它读到上面这一行中的控制词 \it 后就开始选用 italic 字体进行排版，直到它读到表示源文件结束的控制词 \bye 后才知道源文件中缺少表示子体结束的右花括号。这时 TEX 会给出如下信息：

```
This is emTeX (tex386), Version 3.14159 [4a] (no format preloaded)
**try
(try.tex [1] )
(\end occurred inside a group at level 1)
Output written on try.dvi (1 page, 2220 bytes).
Transcript written on try.log.
```

其中

```
(\end occurred inside a group at level 1)
```

表示源文件中缺少表示子体结束的右花括号。虽然这时 TEX 也会建立 .dvi 文件和 .log 文件，但是有可能 TEX 输出的排版结果不是你所期望的。处理这种错误的最简单的方法是把排版结果用打印机打印出来或在屏幕上显示出来，然后你就可以知道应该在什么地方添加右花括号。但若源文件太长，则打印它需要很长时间。这时，可以试着在源文件一半的位置输入表示源文件结束的控制词 \bye，然后再次运行 TEX。如果不再出现这样的错误信息则说明这个错误出现在源文件的后半部分；否则说明错误出现在前半部。如果错误出现在源文件的后半部分，则可以在最后输入的控制词 \bye 的前面输入一个百分号 % (使得这个控制词失效，同时又使你知道它在源文件中的位置)，然后在源文件的 3/4 位置 (即位于 %\bye 和源文件的最后一个 \bye 的中间位置) 输入另一个控制词 \bye。再次运行 TEX 就可以知道错误是位于源文件的前 3/4 部分还是后 1/4 部分。类似地，如果错误在源文件的前 3/4 部分，你可以在源文件的 1/4 位置输入一个控制词 \bye 去估计错误到底是在源文件的前 1/4 部分还是在它的 1/4 到 2/4 部分。按这种方法进行下去，最终会找到错误在源文件中的确切位置。

§1.3.5.2 TEX 的内存限制

有时会遇到这样的错误信息：

```
! TeX capacity exceeded, sorry. ...
```

这样的错误信息是很难处理的。当你的要求超出 TEX 的工作能力之后，它便在给出上面的信息后停止工作。实际上这大多是由于计算机的内存不足或是由于 TEX 软件的内部采用 8 位编码所引起，在当前的计算机配置下，内存已经有 32M 以上，此类错误信息便很少出现了。由于这种错误使 TEX 立即退出运行，我们只能用 §1.3.5.1 中的方法去找出问题所在。

§1.3.5.3 TEX 不认识源文件中表示文件结束的控制词

有时源文件中的错误会使 TEX 跳过一段内容不读而去读后面的内容。因此有时 TEX 会因

为前面的错误而跳过源文件中你已经输入的表示结束的控制词 \bye，这时 TEX 也会给出提示符 *，并且当你为回答这个提示符而输入

```
\bye↩
```

后，它还会给出另一个同样的提示符 *。现在，为了退出 TEX，只好试着输入一个没有定义过的控制词（例如 \HHUH ）来回答提示符 *，如果幸运，TEX 会给出一个 Undefined control sequence 的信息及一个问号？，从而你可以输入 x 去退出 TEX 。如果这个办法不起作用，可以试着按 Ctrl--Z 键、 Ctrl--C 键或 Ctrl--Break 键，这时 TEX 通常会给出一个特殊的“错误信息”以

```
! Iterruption
```

作为第一行。如果这个办法也不起作用，只好重新热启动计算机，即按 Ctrl--Alt--Del 键。

第二章　TEX 定义的格式文件

第一章简单介绍了 TEX 软件的使用，并且使我们能够编写像 try.tex 这样简单的源文件．但是这种一段接一段的课文输出格式不一定能满足你的需要．例如，你可能想要打印一封信件，也可能想要打印一篇论文或者报告，还可能想要打印一本书．为了简化这些排版工作，TEX 提供了一些不同的格式文件，以满足不同的排版需要，本章主要介绍 TEX 中的 plain 格式、vanilla 格式和 Ltr 格式。

§2.1　TEX 中的格式文件与 plain 格式

TEX 提供了几个格式文件，这些格式文件中定义了一些便于理解的 TEX 命令，简化了源文件的输入，使得源文件的编写更加简单。一般地，TEX 的格式文件以 .sty 为扩展名，例如 plain.sty 、vanilla.sty 和 ltr.sty 都是一般的 TEX 软件包附带的格式文件．要使用某种 TEX 格式文件，一般需要在源文件的第一行用带参量的控制词 \input 调入该格式文件．例如若某源文件的第一行是：

```
\input plain.sty
```

则表示该源文件采用的是 plain 格式；若第一行是

```
\input vanilla.sty
```

则表示该文件采用的是 vanilla 格式。要特别注意的是用 \input 命令调入格式文件时，它的后面一定要跟格式文件的全名（即不能省略扩展名 .sty ）．TEX 这样要求的原因是 \input 命令表示调入一个外部文件，如果省略调入文件的扩展名，系统会认为扩展名为 .tex ．因此，TEX 认为

```
\input vanilla
```

和

```
\input vanilla.tex
```

是一样的。

回想第一章中的源文件 try.tex ，我们并没有在它的第一行调入格式文件，那么它采用的是什么格式呢？ TEX 的默认格式为“ plain ”格式，当要使用该格式时，可以省略调入格式文件的命令．因此第一章中建立的源文件采用的是 plain 格式。

§2.2　TEX 中的 vanilla 格式

要想应用 TEX 的默认格式来打印一篇论文的标题，必须键入如下的一些指令：

```
\vglue 24pt plus 12pt minus 12pt
\centerline{\bf Title}
```

这涉及到许多你不想知道的有关 TEX 的和排版方面的术语．所以 TEX 提供了其它一些格式，使你不需要这些专业知识就可以得到各种不同格式的打印输出．本节介绍其中的一个名为“ vanilla ”的格式。下面分几小节介绍 vanilla 格式的特点。

§2.2.1 vanilla 格式的选用

由 2.1 节知道要使用 vanilla 格式，只需要源文件的第一行是

```
\input vanilla.sty
```

即可。

§2.2.2 vanilla 格式与 plain 格式的差别

vanilla 格式与 plain 格式稍有差别，现在把它们之间的不同点列举如下：

1. 在 plain 格式中，符号 @ 可以用键盘上的 @ 键来得到；而在 vanilla 格式中这一符号用控制符 \@ 来表示。
2. 在 plain 格式中，控制符 \{ 和 \} 只能用于数学公式中表示左花括号和右花括号；而在 vanilla 格式中则没有这种限制，这两个控制符可以在正文中使用。
3. 在 plain 格式中，控制符 \,(即反斜杠后紧跟一个逗号) 只能用于数学公式中表示一个小空格 (约为一个字符的 1/2 宽)；而在 vanilla 格式中这个控制符还可以用于普通课文中表示同样的意思。
4. 在 plain 格式中，控制词 \dots 只能用在数学公式中表示省略号；而在 vanilla 格式中该控制词则不受这个限制，并且表示同一个意思 (注意 TEX 不把控制词 \dots 之前的空格打印出来)。

§2.3 vanilla 格式的特点

vanilla 格式提供了一系列的简化输入文章的标题、作者、子标题、定理、证明等的命令，本节主要介绍这些命令。

§2.3.1 vanilla 格式中文章标题和作者姓名的输入

假如你的源文件是以如下的方式开头：

```
\input vanilla.sty
\title An Untitled Masterpiece \\
\rm With an un-Subtitle
\endtitle
\author Your Name \endauthor
```

则 TEX 的打印输出将以如下的格式开头：

An Untitled Masterpiece

With an un-Subtitle

YOUR NAME

注意 TEX 是把控制词 \title 和控制词 \endtitle 之间的字符分两行来打印的，第一行用黑体字打印，第二行用 roman 字体打印，而分行点恰在双反斜杠处。你可能会猜到在 vanilla 格式中，控制词 \title 和控制词 \endtitle 是专为打印文章的标题而设计的，并且可以用双反斜杠 \\ 把标题分成几行来打印。事实正是如此，标题通常用黑体字来打印，而双反斜杠 \\ 表示标

题的分行点。在上面的例子中控制词 \rm 使得第二行标题以 roman 字体输出。如果你想在第一行标题和第二行标题之间多留一点空隙，可以采用如下的方法：

```
\title An Untitled Masterpiece \\

\\ \rm With an un-Subtitle
\endtitle
```

这里，第一个 \\ 和第二个 \\ 之间没有任何字符（除了空格和键入了一次回车键），这表示让 TEX 输出一个空行作为第二行标题：

An Untitled Masterpiece

With an un-Subtitle

YOUR NAME

类似的结构为控制词 \author ...\endauthor，它们是用来打印作者姓名的结构。

§2.3.2 vanilla 格式中节标题和子标题的输入

我们常常需要把一个长文章分成几节来打印，并且在每一节的前面加上一个标题。在 vanilla 中，使用像

```
\heading ... \\ ... \endheading
```

这样的结构输入节标题。例如，源文件中下面一行

```
\heading The First Features\\
of Interest\endheading
```

的打印输出为：

The First Features
of Interest

子标题的输入方法为：

```
\subheading{...}
```

例如下面的一段

```
\subheading{Heading and subheading}
It often helps to divide a long document
into smaller section.
```

的打印输出为：

Heading and subheading. It often helps to divide a long document into smaller section .

注意，在子标题的输入中不能有双反斜杠 \\，也不能在子标题的后面键入句号 (因为 TEX会自动在子标题的后面加上一个句号)。

§2.3.3 vanilla 格式中定理、命题和证明的输入

在数学文章中会有一些定理、命题和定义等等。通常人们喜欢把定理、命题和定义的内容用不同于正文的字体打印出来以示区别。vanilla 中提供了如下的打印定理、命题和定义的输入

格式：

```
\proclaim{label}  ...  \endproclaim
```

其中 label 代表的是 Theorem 、 Corollary 和 Definition 等等，而省略号 ... 则代表的是定理、命题和定义等的内容。例如，我们可以在源文件中输入：

```
\proclaim{Theorem 2.1} Mathematicians
are egotists. \endproclaim
```

相应于上面这一段的打印输出为：

THEOREM 2.1. *Mathematicians are egotists.*

注意在输入中 `Theorem 2.1` 的后面没有句号，而在打印输出时 TEX 自动在 Theorem 2.1 的后面添加了一个句号。还要注意一定要在定理的内容结束处键入控制词 `\endproclaim` ，否则 TEX 会把后面的所有字符都当成定理的内容而用 slanted 字体打印出来。

在 vanilla 中还提供了类似的输入证明内容的方法。证明的输入格式为：

```
\demo{Proof}  ... \enddemo
```

其中省略号 " ... " 表示的是证明过程。例如，我们可以键入

```
\demo{Proof} They are always trying to
make their statements stand out by
printing them in a special type face.
\enddemo
```

相应于上面这一段的输出为：

PROOF: They are always trying to make their statements stand out by printing them in a special type face.

要强调的是我们在输入时并没有在 " Proof " 的后面键入冒号，但打印输出时 TEX 自动在该词的后面添加了一个冒号。

§2.3.4 vanilla 格式的其它特色

vanilla 还提供了打印脚注的功能。打印脚注可以按如下格式输入：

```
\footnote{...}
```

其中省略号 ... 代表脚注的内容。 vanilla 把文章中所有的脚注都按其在文章中出现的先后次序统一编号为 [1] ， [2] ， ... ，并且把每一个脚注的内容都打印在相应页的底部。例如，下面一段源文件

```
Footnotes are eschewed by good writers
\footnote{I try to avoid them.}
nowadays, but if you must have footnotes,
the vanilla style makes them easy to obtain
\footnote{Footnotes are labelled throughout
the file.}.
```

的打印输出为：

Footnotes are eschewed by good writers [1] nowadays, but if you must have foot-

[1] I try to avoid them.

notes, the vanilla style makes them easy to obtain [2].

上面这一段打印输出中有两个脚注，它们的内容分别被打印在本页底部横线下的位置.

脚注也可以用其它的符号来标识，其输入格式为：

```
\footnote"***"{...}
```

其中 *** 代表字符或控制序列。我们可以键入

```
Notice, by the way \footnote{This is
the third footnote},
that \TeX\ will split a
footnote across a page, if necessary
\footnote"\dag"{ No space was
left between the \} and the period, since no
space was wanted between the \dag\ and the
period that follows it. Notice,
by the way, that  \TeX\
will split a footnote across a page,
if necessary (though
some styles might prohibit this.)}.
```

它所对应的打印输出为：

Notice, by the way [3], that TEX will split a footnote across a page, if necessary†.

这一段输出中有两个脚注。第一个脚注是以数字 [3] 作为标记的，这是因为它恰好为本文的第三个脚注，它的内容被打印在本页下面的横线之下；而第二个脚注是以符号 † 作为标记的，该脚注的内容被打印在本页的最底部.

在 vanilla 中，每一自然段的第一行都是在距离左边两个字符的位置开始打印的。如果你想要 TEX 把自然段的第一行从左边沿开始打印，可以用控制词 `\flushpar` 来作为自然段的开头.

另外，还可以使用控制词 `\smallpagebreak` 来增加两个自然段之间的距离，同样还可以使用控制词 `\medpagebreak` 和 `\bigpagebreak` 来达到同一目的。这三个控制词所增加的距离一个比一个大.

还有一个有用的命令是 `\vskip{...}`，它也可以用来增加两个自然段之间距离。如果想为插图留出 10cm 高度的空间，可以键入 `\vskip{10cm}`.

§2.4 vanilla 格式中的空隙、分行和非英文字母

现在我们可以用 vanilla 来建立源文件了。但是为了得到更漂亮的排版效果，还应该了解一些有关空隙和分行的控制词，并且有时还需要打印一些非英文字母，如 œ 和 æ 等等。这一节的前半部分就介绍一些有关空隙和分行的内容，而后半部分介绍一些特殊字符的输入法.

[2] Footnotes are labelled throughout the file.

[3] This is the third footnote

† No space was left between the } and the period, since no space was wanted between the † and the period that follows it. Notice, by the way, that TEX will split a footnote across a page, if necessary (though some styles might prohibit this.)

§2.4.1 vanilla 格式中单词间空隙的控制

为了得到漂亮的排版效果，TEX 经常要调节英文单词之间的空隙。为使右边缘打印整齐而在某一行单词之间添加空隙时，TEX 常把该行中句号的后面添加更大的空隙，以使打印输出的结果更美观。但是，英文的缩写词也是以句号结尾，因而有时 TEX 会把缩写词后面的空隙留得过大。如果在源文件中按如下的方式输入杂志 Proceedings of the American Mathematical Society 的缩写词：

```
Proc.  Amer.  Math. Soci.
```

则 TEX 会认为这是四个不同的句子而把它们之间的空隙打印得太大：

Proc. Amer. Math. Soci.

为修正缩写词后的空隙，需要用到控制符“\.”（反斜杠后跟一个句号）来表示不是作为一个句子结束的句号。我们可以用如下的方法输入上面的缩写词：

```
Proc\. Amer\. Math\. Soci\.
```

相应的输出为：

Proc. Amer. Math. Soci.

还要说明的是 TEX 不把大写字母后的句号当做一个句子的结束（通常一个大写字母后跟一个句号的结构中字母表示姓名的首写字母），从而当缩写词由大写字母组成时，可以按我们习惯的方式输入它。例如，可以键入

```
\TeX\ is the brain-child of D.  E.  Knuth.
```

但这又会出现一个问题。如果一个句子以大写字母结尾，TEX 会把句号后面的空隙留得太小。例如，下面一段输入

```
\TeX\ now run on the PC.
To learn all about \TeX\ read
{\sl The \TeX book}, by D. E. Knuth.
(But skip Appendix D.)
```

的打印输出为：

TEX now run on the PC. To learn all about TEX read *The TEXbook*, by D. E. Knuth. (But skip Appendix D.)

其中“PC.”和“To”之间的空隙就太小。为了修正这个空隙，可以在标点符号的前面键入符号 @（键盘上有这个符号）。可以把上面这段内容按如下的方式输入：

```
\TeX\ now run on the PC@.
To learn all about \TeX\ read
{\sl The \TeX book}, by D. E. Knuth.
(But skip Appendix D.)
```

这时相应的打印输出为：

TEX now run on the PC. To learn all about TEX read *The TEXbook*, by D. E. Knuth. (But skip Appendix D.)

当要打印多重引号时也会遇见空隙太小的问题。在 vanilla 中用符号 @" 表示两个引号之间应有的空隙。例如，下面一段输入

```
When he said `` @" `plain´ or `vanilla´@" ´´
was he talking about yogurt or \TeX@?
```

的打印输出为：

When he said " 'plain' or 'vanilla' " was he talking about yogurt or TEX?

输入时，在符号 @" 的前面或后面键入空格不影响打印输出结果。

§2.4.2 vanilla 格式中分行点的控制

我们把 TEX 打印输出中的每一行的右边缘点称为该行的分行点. 因为 TEX 认为任何单词的后面都可以作为分行点 (除非它的后面紧跟着的是标点符号)，所以它有时会选择我们不喜欢的地方作为分行点。例如，我们不愿意把词组 " Lesson One " 分两行排版，即不愿意把分行点选则在 lesson 的后面。这时可以应用键盘上的 ~ 键。在 vanilla 中，用于某一单词后面的符号 ~ 表示这个单词和下一个单词之间应该有和其它单词间相同的空隙，而且这两个单词之间的位置不能作为分行点。如果你不想把词组 Lesson One 打印成两行，可以按如下的方式输入

```
Lesson~One
```

或

```
Lesson ~ One
```

它们所对应的输出都为：

Lesson One

下面例子中的 ~ 都是应该键入的：

```
Mr. ~Smith
Lesson ~ One
symbols like ~ \dag ~ and ~\ddag.
  from 1 to ~20.
(1)~\TeX isn't perfect.
```

上面一段输入中的第一行里没有用控制符 \. 来代替缩写词后的句号，这是因为符号 ~ 总是表示两个普通的单词之间的空隙.

通常 TEX 认为破折号和连字符的后面可以作为分行点，但是有时我们不希望用它们来作为一行的最后一个字符，例如我们不想让 " pages 32-38 " 中的符号 " - " 被按排在某一行的结尾来打印。这时可以分别使用 @- 、 @-- 和 @--- 来代替 - 、 -- 和 ---，以表示相应的连字符或破折号的后面不能作为分行点。

在 vanilla 中， \newline 、 \linebreak 和 \allowlinebreak 都是用来控制分行点的控制词。 \newline 和 \linebreak 的作用相类似，它们的共同点是把它后面的单词移到下一行的左边缘开始打印，它们的不同之处是 \newline 允许在它前面一个字符的结尾不与右边缘对齐，而 \linebreak 则要求它前面的 (除了空格外的) 第一个字符的结尾与右边缘对齐. 控制词 \allowlinebreak 的意思稍有不同，它与上一章介绍过的控制符 \- 相类似，是表示它所在的位置可以作为分行点 (\- 表示的是它所在的位置是加连字符后可以分行的点)，所以它只能在 TEX 禁止分行的地方才会起作用 (例如破折号之前). 因为当 TEX 发现控制词 \allowlinebreak 不是位于某一行的结尾时就不另起一行开始打印，所以人们在 TEX 禁止分行的地方更喜欢使用

它而不用 \linebreak 和 \newline 。注意这三个控制词都不能像控制符 \- 一样用在单词的中间。例如，下面的输入

```
There are also two other rather \allowlinebreak
special commands to control lilne breaks.
These control sequences are analogous to the
line breaking control sequences, but there
is an difference.

There are also two other rather \linebreak
special commands to control lilne breaks.
These control sequences are analogous to the
line breaking control sequences, but there
is an difference.

There are also two other rather \newline
special commands to control lilne breaks.
These control sequences are analogous to the
line breaking control sequences, but there
is an difference.
```

所对应的输出为：

There are also two other rather special commands to control lilne breaks. These control sequences are analogous to the line breaking control sequences, but there is an difference.

There are also two other rather special commands to control lilne breaks. These control sequences are analogous to the line breaking control sequences, but there is an difference.

There are also two other rather
special commands to control lilne breaks. These control sequences are analogous to the line breaking control sequences, but there is an difference.

由此可见 \allowlinebreak 、 \linebreak 和 \newline 所产生的输出效果之间的差别。

§2.4.3 vanilla 格式中重音符和特殊字符的打印

现在分别介绍 TEX 中所提供的一些打印重音符和非英文字母的方法。下面的例子中，左边列出的是控制序列作用于字母 o ，而右边是相应的打印输出：

\`o	ò	\'o	ó
\^o	ô	\"o	ö
\~o	õ	\u o	ŏ
\v o	ǒ	\H o	ő
\B o	ō	\b o	o̲
\D o	ȯ	\d o	ọ
\c o	ǫ	\t oo	o͡o

上面这些控制序列也可以作用于其它字母，但是当它们作用于字母 i 和 j 时，这两个字母应该用 ı 和 ȷ 来代替。而在 vanilla 中，ı 和 ȷ 可以分别由控制词 \i 和 \j 得到。例如，下面的输入

```
Don't be too na\"\i ve about accents.
```

所对应的打印输出为：

Don't be too naïve about accents.

我们可以在任何字体中应用上述控制序列打印重音符。例如，{\bf\"o} 的打印输出为 **ö**，而 {\sl\^A} 的打印输出为 *Â*。

TEX 还提供了下列特殊字母：

\oe	œ	\OE	Œ
\ae	æ	\AE	Æ
\aa	å	\AA	Å
\o	ø	\O	Ø
\l	ł	\L	Ł
\ss	ß		

Voilà! 现在你可以打印一个 *œuvre en français*。

另外，在 vanilla 格式中还可以打印倒问号 ¿ 和倒感叹号 ¡，它们的输入方法是在普通的问号和感叹号的后面键入左单引号：

!`	¡
?`	¿

§2.5 vanilla 格式中的数学公式

在这一节里我们介绍 vanilla 格式中数学公式的输入方法。为了使你更好的理解这些方法，我们在课文中给出了一些练习，其答案可以在本节的 2.5.3 段中找到。

§2.5.1 vanilla 格式中数学公式的表示方法

由于计算机毕竟只是一台机器，因此必需采用一种特殊的处理方法来使计算机明白你要建立一个公式。TEX 选择了美元符号 $ 来表示一个数学公式的开始和结束。通常，数学公式可以出现在正文中，也可以单独列出。我们把出现在正文中的数学公式简称为正文公式，把单独列出的数学公式简称为单列公式，而把正文公式和单列公式统称为数学公式。这一节中，我们分别介绍正文公式和单列公式、上标和下标以及分式和二项式系数的输入方法。

§2.5.1.1 vanilla 格式中正文里的数学公式

当在源文件中键入 $...$ （其中省略号 ... 表示一段字符）时，TEX 就把两个 $ 之间的字符看作正文公式。TEX 在输出时把数学公式之中的英文字母用 italic 体来打印，并且总是忽视数学公式中的空格。因为 TEX 把非空行结尾的回车键看作空格，所以可在源文件中把一个数学公式分两行来输入。在数学公式中，连字符 “ - ” 代表减号 “ − ”。斜杠 “ / ” 代表分号 “ / ”；而键盘上的 | 键代表竖线 | (常用它来表示绝对值)。例如，下面一段文字

The | key will become a solid vertical line, this is used in formulas like $f(x) = |x| - l$.

在源文件中是按如下方式输入的：

```
The | key will become a solid vertical line,
this is used in formulas
like $f(x)=|x|-l$.
```

注意，在源文件中数学公式里的小写字母 l 有一点像数字 1，而这个字母的输出则变为 italic 体 l (它不容易和数字 1 混淆)。

为了便于察看源文件中的数学公式，你可以在数学公式中多插入一些空格，但是不能随便键入花括号。例如，当 TEX 读完 $y=x{+}1$ 时，它会认为你想用某种特殊方式处理加号 +，从而在打印输出时加号两边的空隙就变小 (比较它所对应的输出 $y = x{+}1$ 和 $y=x+1$ 所对应的输出 $y = x + 1$)。

数学公式中最常用的符号 +、−、/、<、>、= 和 | 可以用键盘上相应的键输入，但是数学上常用的其他符号却需要用控制序列来表示。在第 69 页的 §2.9 中列出了一些表示数学符号的控制序列。下面是几个常用的数学符号的定义：

定义	\le	\ge	\ne	\in	\notin	\infty
符号	$\le$	$\ge$	$\ne$	$\in$	$\notin$	∞

由于希腊字母常被用来作为特殊的数学符号，我们也把表示它们的控制词列入到第 69 页的第 §2.9.1 小节中。例如 α、β 和 γ 分别表示头三个希腊字母 α、β 和 γ，而 Γ 则代表大写的希腊字母 Γ (注意 \Alpha 和 \Beta 都不是 TEX中的控制词，这是因为希腊字母 α 和 β 的大写很像 roman 体英文字母 A 和 B)。

注意不要把正文中的标点符号或文字打在正文公式里，否则这些符号或文字后应该有的空隙可能会消失。例如，下面一段源文件

```
For all $a$ and $b$ we have $a<b, a=b$,
or $a>b.  So <$ is a partial ordering.
```

对应的的输出为：

For all a and b we have $a < b, a = b$, or $a > b.So <$ is a partial ordering.

因为 TEX 把输入中的 `$a<b, a=b$` 看作一个数学公式，所以它不把逗号后面的空格打印出来。同样的原因使得输入 `$a>b. So <$` 中的所有空格在输出中消失，并且把单词 So 用 italic 字体打印出。前面介绍过控制符 \ 代表一个空格，它也可以在数学公式中使用，那么是不是可以用如下的方法输入上一段内容？

```
For all $a$ and $b$ we have $a<b,\ a=b$,
or $a>b.\  So\ <$ is a partial ordering.
```

答案是肯定的．这一段所对应的输出为：

For all a and b we have $a < b$, $a = b$, or $a > b$. *So* $<$ is a partial ordering.

注意，单词 So 是用 italic 字体 (而不是用与正文相同的 roman 字体) 打印出来的．

练习 1. 用 TEX 打印下面一段内容：

We have $|\alpha(x,y) - \beta(x,y)| \le |10\alpha - l| < |\beta|$, for all $\alpha \in A$, $\beta \in B$ and $\gamma \in \Gamma$.

But if $\gamma \notin \Gamma$, then $f(\alpha,\beta) \ge \{\gamma\} \ne -\infty$.

即使你能够熟练地运用 TEX 打印数学公式，偶尔也会漏掉几个表示数学公式开始或结束的符号 $。这时 TEX 将把你的源文件中的一部分文字当成数学公式，从而会打印出奇怪的结果．幸运的是 TEX 在编译源文件时能够发现这类错误，提示你应该插入或删除 $．TEX 知道像 `\le`、`\infty` 和 `\alpha` 这样的一些特殊控制序列只能用在数学公式中，当它在编译源文件时遇到这样的控制序列出现在正文时就会给出错误信息，告诉你它打算插入表示数学公式的符号 $．这时可以键入 x 退出 TEX 去纠正源文件中的错误，或者击回车键让 TEX 继续运行，然后回到源文件中统一修改所有的错误。

TEX 不允许在数学公式中有空行和控制符 `\par` 出现。当 TEX 发现这种情况出现时，它就会给出错误信息，说你忘了在自然段结束之前结束数学公式。

现在可以在你的源文件中输入下面的一段：

```
We forgot to put a dollar sign at
the beginning of the formula \alpha  +
\beta$ in this paragraph.

And we forgot to put one at the end
of the formula $x+y in this one.

We'll probably get some error messages.
```

然后运行 TEX 对你的源文件进行排版，看看 TEX 会产生出现些什么错误信息．这时 TEX 也会产生 .dvi 文件，它的打印结果如下：

We forgot to put a dollar sign at the beginning of the formula $\alpha + \beta$ in this paragraph.

And we forgot to put one at the end of the formula $x + yinthisone$.

We'll probably get some error messages.

TEX 能很好地在一个英文单词中确定 (通过加连字符) 可以分行的点，相比之下它在一个正文公式中寻找分行点的能力就差得多。当正文公式出现在某一行的结尾时，TEX 可能因为找不到合适的分行点而把这个公式的结尾作为分行点，同时给出信息说这一行太长。这时就需要你用控制词 `\mathbreak` 告诉 TEX 在公式中的什么地方分行，或者用控制词 `\allowmathbreak` 为 TEX 指出公式中可以作为分行点的位置。TEX 规定在正文中使用的控制词 `\linebreak` 、`\newline` 和 `\allowlinebreak` 都不能在数学公式中使用，而控制词 `\mathbreak` 和 `\allowmathbreak` 也不能在正文中使用。

§2.5.1.2 vanilla 格式中单独列出的数学公式

在数学文章中，重要的公式和很长的公式都应该单独列出。在 TEX 中用符号 `$$` 表示一个需要单独列出的数学公式的开头和结尾。当你键入

```
$$1+1=2$$
```

时，TEX 就把 `1+1=2` 作为单列公式打印如下：

$$1+1=2$$

TEX 读到两个双美元符号 `$$` 之间的内容后就把它们安排在下一行的中间位置作为单列公式来打印。不管单列公式有多长，TEX 都把它们在一行中打印出来 (除非你用将在第四节中介绍的方法告诉它这是多行公式)，并且还不允许在它中使用控制词 `\mathbreak` 和 `\allowmathbreak` 来分行。同对待正文中的数学公式一样，TEX 也不允许在单独列出公式的输入中有空行或者控制词 `\par` 的出现，也忽视在单独列出公式中键入的空格。

要特别注意单列公式后面的标点符号，它应该在表示公式结束的双美元符号 `$$` 之前输入。而在正文公式后的标点符号则要把标点符号放在 `$` 之后。例如，当我们想要打印出如下的一段文字：

We've finally managed to prove that

$$4!=12.$$

Let's celebrate!

时，应该输入：

```
We've finally managed to prove that
$$4!=12.$$
Let's celebrate!
```

而不应该把句号与双美元符号 `$$` 的位置互换，即键入

```
We've finally managed to prove that
$$4!=12$$.
Let's celebrate!
```

这时对应的输出是：

We've finally managed to prove that

$$4!=12$$

Let's celebrate!

. Let's celebrate!

其中的句号被打印在单词 Let's 的前面，这不是我们所希望的。

练习 2. 用 TEX 打印下面一段内容：

We have

$$|\alpha(x,y)-\beta(x,y)| \le |10\alpha - l| < |\beta|,$$

for all $\alpha \in A$, $\beta \in B$ and $\gamma \in \Gamma$. But if $\gamma \notin \Gamma$, then

$$f(\alpha,\beta) \ge \{\gamma\} \ne -\infty.$$

§2.5.1.3 vanilla 格式中数学公式里的上标和下标

上标和下标是数学公式中经常出现的。在源文件的数学公式中，用键盘上的符号 ^ 和 _ 分别表示其后面的第一个 (非空格) 字符或子体为前面一个 (非空格) 字符或子体的上标和下标. 例如：

`$x^2$`	x^2
`$x^\alpha$`	x^α
`$2^x$`	2^x
`$x_y$`	x_y
`$x^2y^2$`	x^2y^2
`$x^2_3$`	x^2_3
`$x^{2y}$`	x^{2y}
`$2^{32}$`	2^{32}

下面列出的是更复杂的上下标表示方法：

`$2+{}_np^{(m)}_{ij}$`	$2+{}_np^{(m)}_{ij}$
`$2^{\{\alpha\}}$`	$2^{\{\alpha\}}$
`$(x^2)^3$`	$(x^2)^3$
`$f_+$`	f_+

上面的第一行说明空子体 {} 和 _(或 {} 和 ^) 可以用来打印一个字符的左下 (或上) 标；第二行中的花括号是上标的一部分；第三行中 3 成为右小括号的上标；第四行中加号成了下标。

也可以分别用控制词 \sp 和 \sb 来代替键盘上的符号 ^ 和 _，这两个控制词对于键盘上没有 ^ 键和 _ 键的使用者来说非常重要。当你使用 \sp 和 \sb 时，不要忘记在它们的后面键入空格。

练习 3. 把下面的一段文字编写成源文件：

If $a_1 \le 5$, $a_2 \le 6$ and $\alpha \le 7$, then

$$(a_1+a_2)^{2\alpha} \le 11^{14} = (11^7)^2.$$

下面的这些例子显示了多重下标的输入方法：

```
$a^{b^c}$          a^{b^c}
$a^{b^{c+1}}$      a^{b^{c+1}}
$2^{(2^x)}$        2^{(2^x)}
$x_{y_2}$          x_{y_2}
$x_{y^2}$          x_{y^2}
```

练习 4. 给出 $2^{2^{2^{2^{2^{2^{2}}}}}}$ 的输入方法。

当你的源文件中有类似于 a^b^c 和 a_b_c 的输入时 TEX 会给出错误信息，但是可以有下面例子中的输入方法：

```
$A^a_b$                 A^a_b
$z_{ij}^*$              z_{ij}^*
$Z_{y^a_ b}^{z_c^d}$    Z_{y^a_ b}^{z_c^d}
```

利用空子体 {} 可以产生特殊的上下标打印效果，例如：

```
$x_i{}^2$               x_i{}^2
$R_i{}^{jk}{}_l$        R_i{}^{jk}{}_l
```

在数学公式中，可以直接用右引号 '(通常在表示函数的导数时用) 作为上标：

```
$f'$                    f'
$f'_2$                  f'_2
$y_1'+y_2''+ y_3'''$    y_1'+y_2''+y_3'''
```

符号 ′ 是作为上标的撇号 ′(在源文件中控制词 \prime 表示这一符号)，所以用 $f^\prime$ 和 $f^{\prime\prime}$ 也可以得到 f' 和 f''。显然在这种情况下人们更喜欢用左单引号，但是有时必须使用 \prime：

```
$g^{\prime2}$    g^{\prime2}
```

如果你键入的是 g'^2，TEX 就会给出错误信息；如果你键入的是 ${G'}^2$，TEX 相应的输出则变为 G'^2。

练习 5. 如何在源文件中输入下面的数学公式？

$$f^{(n+2)} = [f^{(n+1)}]' = [f^{(n)}]''$$

在极特殊的情况下输入上下标会产生奇怪的现象。例如，当你想要打印 $A_{\neq}$ 和 $\Gamma_{\notin}$ 时必需输入 $A_{\ne}$ 和 $\Gamma_{\notin}$。如果你忘记了键入花括号，TEX 会给出错误信息或者打印出的不是你想要的符号。记住，当输出的上下标出现这类问题时，你把相应的上下标用花括号括起来就可以解决问题。

§2.5.1.4 vanilla 格式中数学公式里的分式和二项式系数

像 x/y 和 $(a+b)/2$ 这样的公式常出现在正文中，但是在单列公式中却常用如下形式的分式：

$$\frac{n+1}{n+3}.$$

它的的输入方法为：

```
$$\frac {n+1}{n+3}$$
```

控制词 \frac 带有两个参量，第一个参量为分子，第二个参量为分母。

如果你在正文中键入 `$\frac23$` 可以得到 $\frac23$。也可以在分式中打印分式：

```
$$\frac x{1+\frac x2}$$
```
$$\frac{x}{1+\frac{x}{2}}$$

```
$$\frac {\frac x2+1}2$$
```
$$\frac{\frac{x}{2}+1}{2}$$

如果你觉得上面的例子中的小分式 $\frac{x}{2}$ (这种尺寸常在正文中使用) 打印成 $\frac{x}{2}$ 更好看，可以使用控制词 \dfrac 来代替 \frac 来得到普通尺寸的分式：

```
$$\frac x{1+\dfrac x2}$$
```
$$\frac{x}{1+\dfrac{x}{2}}$$

练习 6. 在源文件中如何输入下面的分式？

$$\frac{\frac{x}{2}+1}{2}$$

偶尔你会在单列公式中需要打印正文中使用的小分式，这时可以使用控制词 \tfrac 。例如，`$$\tfrac 34$$` 的输出为

$$\tfrac34$$

更一般地，\dsize 和 \tsize 是用来放大或缩小数学公式尺寸的控制词，它们都不是带参量的控制词，其用法和控制词 \it 相类似。如果你想放大 (缩小) 一个公式中的某一部分，一定要把这一部分用花括号括起来。例如：

```
$$\frac x{1+{\dsize\frac x2}}$$
```
$$\frac{x}{1+{\dsize\frac{x}{2}}}$$

在数学公式中常用到二项式系数 $\binom{a}{b}$，它在单独列出的公式中为：

$$\binom{a}{b}$$

上面的二项式系数在数学公式中用控制词 \binom ab 表示，当它出现在正文公式中时 TEX 用小尺寸打印，当他出现在单列公式中时 TEX 用大尺寸打印。和分式类似，也可以用 \dbinom 和 \tbinom 表示选用大尺寸或者小尺寸来打印二项式系数。

输入 `$$\frac{\binom nk}2$$` 的输出为：

$$\frac{\binom{n}{k}}{2}$$

而输入 `$$\binom n{\frac k2}$$` 的输出为：

$$\binom{n}{\frac{k}{2}}$$

练习 7. 在源文件中输入下面三个单列公式：

$$\frac{\binom{n}{k}}{2}$$

$$\binom{n}{k/2}$$

$$\binom{n}{\frac{1}{2}k}$$

§2.5.2 vanilla 格式中常用数学符号的输入

我们这一段中介绍常用数学符号在源文件中的表示方法。

§2.5.2.1 vanilla 格式中求和符号和积分符号的输入

数学家们常约定一些大写的希腊文字母表示特定的意思。例如， `$\Sigma$` 在源文件中代表大写的希腊字母 Σ，但是数学家却常用符号 $\sum$ (它比大写的希腊字母 Σ 稍大一些) 表示求和号。在数学公式中的求和号通常都是带有上下标的，例如正文公式 $\sum_{i=1}^{n} a_i$ 。在单列公式中的求和号更大一些，其上下标打印的位置也不同，例如

$$\sum_{i=1}^{n} a_i.$$

TEX 使用控制词 `\sum` 表示求和号。 TEX 把出现在正文公式里的求和号用较小的尺寸打印，而把单列公式中的求和号用较大的尺寸打印。下面的一段源文件

```
the formula $\sum_{i=1}^n a_i$ in text.
```

所对应的输出为：

the formula $\sum_{i=1}^{n} a_i$ in text.

而输入

```
the formula $$\sum_{i=1}^n a_i$$ in
display math   mode.
```

的打印输出为：

the formula

$$\sum_{i=1}^{n} a_i$$

in display math mode.

要想在正文公式中使用像 $\sum_{i=1}^{n} a_i$ 这样的大求和号可以使用 `{\dsize\sum ...}` 这样的结构输入，例如 `$\dsize\sum_{i=1}^n a_i$` 所对应的输出为 $\sum_{i=1}^{n} a_i$ 。

有时求和号还带有多重下标：

$$\sum_{\substack{0<i<m\\0<j<n}} P(i,j)$$

我们知道当键盘上没有 _ 键时，可以用控制词 \sb 代替它。与此类似，在求和号中的多行下标是用 \Sb ... \\ ... \endSb 这样的结构来输入的，其中 \\ 是下标的分行点。上面的求和号在源文件中的输入为：

```
$$\sum\Sb 0<i<m\\0<j<n\endSb P(i,j)$$
```

类似地可以用 \Sp ... \\ ... \endSp 这样的结构输入多重上标。在附录 A 中列出的其它一些类似于 \sum 的控制词也用上述方法输入多重上下标。

练习 8. 试在源文件中输入单列公式

$$0 = (1 + -1)^n = \sum_{j=0}^{n} (-1)^j \binom{n}{j}$$

和

$$\sum_{\substack{1\le i\le p\\1\le j\le q\\1\le k\le r}} a_{ij} b_{jk} c_{ki}$$

另一个在数学公式中常用的符号是积分号 $\int$，在源文件中的数学公式里用控制词 \int 表示它。单列公式中的积分号比正文公式中的积分号稍大一些，但是它们的上下标的位置却是相同的：

```
$\int_a^b$
```
$\int_a^b$

```
$$\int_a^b$$
```
$$\int_a^b$$

还有其它一些积分符号：

```
$$\oint$$
```
$$\oint$$

```
$$\iint$$
```
$$\iint$$

```
$$\iiint$$
```
$$\iiint$$

```
$$\iiiint$$
```
$$\iiiint$$

有时需要像求和号的多重下标一样把积分区间放在积分号的下面，这可以使用控制词 \limits：

```
$$\iint\limits_S\omega$$
```
$$\iint\limits_S \omega$$

积分符号通常出现在如下的这类数学公式之中：

$$\int_a^b f(x)\,dx$$

这个公式中的 $f(x)$ 和 dx 之间有一点空隙。不幸的是 TEX 不能在类似的地方自动添加这样的空隙，只有它看见表示小空隙的控制符 \,(反斜杠后跟一个逗号) 时才给添加这样的空隙。上面的数学公式是用如下的方法在源文件中输入的：

```
$$\int_a^b f(x)\,dx$$
```

在微分公式中也应该有这样的小空隙，例如在源文件中数学公式

$$dx\,dy = r\,dr\,d\theta$$

的输入为：

```
$$dx\,dy=r\,dr\,d\theta$$
```

练习 9. 在源文件中输入单列公式

$$\int f(x)\,dx = \int f(x)\frac{du}{dx}$$

和

$$\iiint_M d\omega = \iint_{\partial M} \omega$$

(其中符号 ∂ 用 \partial 来表示，而小写希腊字母 ω 则用控制词 \omega 来输入。)

§2.5.2.2 vanilla 格式中数学公式里的可变尺寸符号

在数学公式中有一些符号的尺寸要随着公式的大小而改变，这一段中介绍这类符号。当你用 \frac 打印分式的时候，分式中的直线会随着分式的长短而改变。还有一些控制词有类似的功能，例如：

```
$\underline{x+y}$
```
$\underline{x+y}$

```
$\overline{x+y+z}$
```
$\overline{x+y+z}$

```
$\overrightarrow{x+y}$
```
$\overrightarrow{x+y}$

根号是数学上很常用的的符号，在 TEX 中它用 \sqrt 表示。例如：

```
$\sqrt2$
```
$\sqrt2$

```
$$\sqrt{\frac ab +1}$$
```
$$\sqrt{\frac ab +1}$$

更复杂一些的开方号是用 \root ...\of ... 的格式输入的：

```
$$\root 3\of {x+y}$$
```
$$\root 3\of {x+y}$$

打印像

$$\left(\sqrt{\frac ab+(x+y)^2}+1\right)^2$$

这样的数学公式需要用大一点的圆括号。在源文件中，上面的公式是按如下的格式输入的：

```
$$
\left( \sqrt{\frac ab+(x+y)^2}+1\right)^2
$$
```

在上例中的“ \left(”和“ \right) ”分别表示这两个圆括号在公式中是可变尺寸括号，TEX 读到它时会根据公式的大小而选取合适的圆括号。类似地，你可以把控制词 \left 和 \right 用在方括号、花括号和绝对值号等符号的前面使得这些符号成为可变尺寸符号：

```
$$\left[\frac AB\right]$$
```
$$\left[\frac AB\right]$$

```
$$\left\{\frac AB\right\}$$
```
$$\left\{\frac AB\right\}$$

```
$$\left|\frac AB\right|$$
```
$$\left|\frac AB\right|$$

```
$$\left\|\frac AB\right\|$$
```
$$\left\|\frac AB\right\|$$

我们可以用输入

```
$$x\in\left(\frac ab,
    \frac cd\right]$$
```

来打印公式

$$x \in \left(\frac{a}{b}, \frac{c}{d}\right]$$

但是要特别注意的是 \left 和 \right 必需在数学公式中成对出现，因此像

$$\frac{dx^2}{dx}\bigg|_{x=a} = 2a$$

这样的公式需要输入一个“ \left. ” 产生一个空的左括号：

```
$$\left.\frac{dx^2}{dx}
\right|_{x=a}=2a$$
```

当然，也可以使用“ \right. ” 产生一个空的右括号，例如公式

$$x^2 \Big/ \frac{c+1}{d}$$

在源文件中的输入为：

```
$$x^2\left/ \frac{c+1}d \right.$$
```

TEX 可以根据公式的需要改 \left 和 \right 之后的括号、绝对值号以及符号 ‖ 的大小，并且这种放大没有什么限制。但是数学公式中的除号 / 却不能用 \left ... \right 无限度的放大，当你的数学公式很大时，TEX 只能用最大尺寸的除号来打印。

在数学公式中分别用控制词 \langle 和 \rangle 表示尖括号“〈”和“〉”，也可以用 \left\langle 和 \right\rangle 使它们成为可变尺寸符号。

练习 10. 在源文件中输入下面的单列公式：

$$ax^2+bx+c=a\left(x^2+\frac{b}{a}x+\frac{c}{a}\right)=a\left[\left(x+\frac{b}{2a}\right)^2+\left(\frac{c}{a}-\frac{b^2}{4a}\right)\right]$$

虽然可以使用 \left 和 \right 得到可变尺寸符号，TEX 还是需要一些控制序列表示一些特殊尺寸的符号。例如使用控制词 \bigl 和 \bigr 可以得到比常用尺寸稍微大一点的符号。在源文件中

```
$\bigl(x-s(x)\bigr)\bigl(y-s(x)\bigr)$
```

的输出为：

$$\bigl(x-s(x)\bigr)\bigl(y-s(x)\bigr)$$

在上面的这个例子中，如果分别用 \left 和 \right 来代替 \bigl 和 \bigr，就会得到下面的输出：

$$(x-s(x))(y-s(x))$$

其中的圆括号都被打印成一样大.这是因为用普通的圆括号就可以把该公式在 \left(和 \right) 之间的部分包括住。

有时用 \left 和 \right 后又会使相应的符号变的太大。例如，输入

```
$$\left(\sum_{i=1}^na_i\right)^2$$
```

的输出为：

$$\left(\sum_{i=1}^{n} a_i\right)^2$$

通常把上面公式中的圆括号打印得稍微小一点更好看：

$$\biggl(\sum_{i=1}^{n} a_i\biggr)^2$$

它的输入方式是把前面输入中的 \left 和 \right 分别改成 \biggl 和 \biggr 而得到的：

```
$$\biggl(\sum_{i=1}^na_i\biggr)^2$$
```

类似地，控制词 \Bigl 和 \Bigr 用在左右括号前将把括号的高度放大到括号 \big(的 1.5 倍，而 \Biggl 和 \Biggr 用在左右括号前将把括号的高度放大到 \big(的 2.5 倍。

练习 11. 试打印下面的公式：

$$m^{\star}\left(A\cap\left[\bigcap_{i=1}^{n}E_i\right]\right)=\sum_{i=1}^{n}m^{\star}(A\cap E_i)$$

其中符号 ∪ 和 ∩ 分别由控制词 \cup 和 \cap 表示，而大尺寸的这两个符号分别由 \bigcup 和 \bigcap 表示。

§2.5.2.3 vanilla 格式中数学公式里特定函数和符号的表示

有时在数学公式中也需要用普通的 roman 字体打印一些特定函数和符号的名称，例如“ sin ”、“ cos ”、“ log ”、“ max ”和“ lim ”等，在 TEX 中这些符号分别用控制词 \sin、\cos、\log、\max 和 \lim 来表示：

`$\sin^2x+\cos^2x=1$`	$\sin^2 x+\cos^2 x=1$
`$\sin2x=2\sin x\cos x$`	$\sin 2x=2\sin x\cos x$
`$\sin(x+\pi)=-\sin x$`	$\sin(x+\pi)=-\sin x$
`$\log_2x=(\log_2e)(\log x)$`	$\log_2 x=(\log_2 e)(\ln x)$

\max 和 \lim 的用法和 \sum 类似，它们通常都带有上下标，并且在正文公式和单列公式中上下标打印的位置也不同。

```
$\max _{1<i<N} A_i$
```
$\max _{1<i<N} A_i$

```
$$\max_{1<i<N}A_i$$
```
$$\max_{1<i<N}A_i$$

```
$$\lim_{x\to\infty}f(x)$$
```
$$\lim_{x\to\infty} f(x)$$

下面列出 TEX 中所有的这类控制词的名称，其中前面标有 (L) 的表示这个控制词可以带有像 \sum 一样的上下标。

\arccos		\csc		\ker	(L)	\min	
\arcsin		\deg		\lg	(L)	\Pr	
\arctan	(L)	\det	(L)	\lim		\sec	
\arg		\dim	(L)	\liminf		\sin	
\cos		\exp	(L)	\limsup		\sinh	
\cosh	(L)	\gcd		\ln	(L)	\sup	
\cot		\hom		\log		\tan	
\coth	(L)	\inf	(L)	\max		\tanh	

练习 12. 在源文件中输入下面的单列公式：

$$\frac{\max_{1<n<m} \log_2 P_2}{\lim_{n\to 0} \frac{\sin x}{x}}$$

在数学公式中，除了特定函数和符号的名称以外还有一些普通的单词要用 roman 字体打印，例如：

$$g(x) = f(x + \text{constant}) + e^{\text{constant}}$$

如果你想要在数学公式中像正文一样打印几个单词，可以使用带参量的控制词 \text，并且把你想打印的单词作为这个控制词的参量．上面的公式是按如下的方法输入的：

```
$$g(x)=f(x+\text{constant})
  +e^{\text{constant}}$$
```

注意，\text 是带参量的控制词，一定要把它的参量用花括号括上．如果把上面的输入改成

```
$$g(x)=f(x+\text constant)
  +e^{\text constant }$$
```

则输出变为：

$$g(x) = f(x + constant) + e^{Constant}$$

控制词 \text 是让 TEX 把它的参量按正文的方式打印，你可以像在正文中一样在 \text 的参量中使用控制词 \it 、\sl 和 \bf，甚至你还可以在参量中使用 $...$ 这样的结构打印正文公式。输入

```
$$\Gamma(n)=(n-1)!
\text{ when $n$ is an integer}\tag{2.1}
```

```
$$
```

所对应的输出是（其中控制词 \tag 的使用方法见下一小段）：

$$\Gamma(n) = (n-1)! \text{ when } n \text{ is an integer} \tag{2.1}$$

注意这个公式中在 when 的前面打印了一个空格，这是因为输入时在 { 和 when 之间键入了一个空格。如果输入的是

```
$$\Gamma(n)=(n-1)!
\text{when} n \text{is an integer}
$$
```

则输出为：

$$\Gamma(n) = (n-1)!\text{when}n\text{is an integer}$$

其中 when 前面的空格以及 n 两边的空格都消失了，这是因为 TEX 总是忽视数学公式中的空格。

在类似于 (2.1) 这样的单列公式中有一个等式和一段说明这个等式的文字，人们习惯在等式和说明文字之间多加一些空隙，为此可以使用控制词 \quad 和 \qquad 。TEX 规定在数学公式中的 \quad 和 \qquad 分别给出约为 2 个字符宽的和约为 4 个字符宽的空隙。因此 (2.1) 式最好按如下的方式输入：

```
$$\Gamma(n)=(n-1)! \qquad
\text{ when $n$ is an integer}\tag{2.2}
$$
```

它所对应的输出为：

$$\Gamma(n) = (n-1)! \qquad \text{when } n \text{ is an integer} \tag{2.2}$$

练习 13. 输入下面的单列公式：

$$\sum_{l \text{ odd}} \binom{n}{l} = 2^{n-1} \qquad \text{for } n \geq 1.$$

§2.5.2.4 vanilla 格式中单列公式的编号

我们已经介绍了输入正文公式和单列公式的方法，这一段中介绍单列公式的编号。人们常喜欢给单列公式编号，例如下面的公式

$$1 + 2 = 3 \tag{2.3}$$

就有编号 (2.3)。在 vanilla 格式中，单列公式的编号是在控制词 \tag 的后面输入。上面的公式是按如下的方法输入的：

```
$$1+2=3\tag{2.3}$$
```

TEX 把单列公式中 \tag 后的字符作为普通的正文打印在公式的左（或右）边的圆括号之中作为编号。首先要注意的是编号的圆括号是 TEX 自动添加上的，不是你键入的；其次要注意 tag 后的连字符 - 和 -- 将分别打印成正文中的连字符和破折号，而不是数学公式中的减号和双减号。

有时候，因为公式的编号与公式靠的太近而显得特别难看。常用的处理方法是把公式的编号单列一行。所以当 TEX 发现这种情况发生时，它就会把公式下移一行来打印，而编号还打印在原来那一行的左（或右）边。如果你觉得这样不好看，可以用在 2.5.2.6 段中介绍的方法把公式分两行打印出来。

还需要强调的是带编号单列公式中标点符号的输入。如果在公式的尾部有标点符号，一定要把它放在 \tag 之前输入，否则 TEX 会把标点符号当做公式的编号打印。例如，输入

```
$$1+2=3 , \tag{2.4}$$
We proves that \dots
```

的输出为：

$$1+2=3, \tag{2.4}$$

We proves that ...

而

```
$$1+2=3 , \tag4.6 ,$$
We proves that \dots
```

的输出则把 (4.6 ,) 作为上面公式的编号。

§2.5.2.5 vanilla 格式中单列公式的对齐

有时会接连有两个或更多的单列公式，这就要把这些公式沿等号或其它的某一符号对齐才好看：

$$\begin{aligned} f(x) &= x^2 \\ f(x+y) &= (x+y)^2 \end{aligned}$$

这两个公式在 vanilla 格式中的输入为：

```
$$
\align
f(x)&=x^2\\
f(x+y)& =(x+y)^2
\endalign
$$
```

其中双反斜杠 \\ 是分行命令，此处表示第一个公式的结束和第二个公式的开始。只有分行命令是不够的，还必须告诉 TEX 你希望把两个公式在何处对齐。在 TEX 中对齐的命令用 & 表示。这里是表示两个公式在等号 " = " 的前面对齐。要注意的是 \endalign 上面的一行，无需在该行的结尾处键入双反斜杠 \\ 。这里的 \align ...\endalign 结构以及随后的 \cases ...\endcases 结构十分相近，都是常出错误的地方，需要特别留心。

练习 14. 输入下面三个公式：

$$\alpha_4 = \sqrt{\tfrac{1}{2}}$$
$$\alpha_8 = \sqrt{\tfrac{1}{2}+\tfrac{1}{2}\sqrt{\tfrac{1}{2}}}$$
$$\alpha_{16} = \sqrt{\tfrac{1}{2}+\tfrac{1}{2}\sqrt{\tfrac{1}{2}+\tfrac{1}{2}\sqrt{\tfrac{1}{2}}}}$$

如果觉得练习 14 中的几个公式之间的垂直距离太近了，可以在键入表示公式结束的双反斜杠 \\ 后键入

```
\vspace{...}
```

表示这个公式和下一个公式之间增加“ ... ”单位的距离，其中“ ... ”可以是“ 0.1 in ”（表示 0.1 英寸）或“ 2 cm ”（表示 2 厘米）或“ 6pt ”（表示 6 点）等等。如果你不能很好地掌握这些度量单位也没有关系，可以使用一种特殊的度量单位 \jot 键入

```
\vspace{1\jot}
```

将在两个公式中添加一点距离，这也许正是你想要增加的距离。如果添加的距离还不合适，你可以用 {1.5\jot} 或 {.75\jot} 等再试一试。使用 \jot 作为度量单位有一个优点，它会随着文章的输出版面的尺寸改变而改变，所以当你想放大或缩小打印输出的版面时就不用再调整单列公式之间的距离了。

注意，如果在数学公式中使用了 \align ...\endalign 的结构，并且由于公式又太宽而打印不下公式的编号时，TEX 会把编号和相应的公式分两行打印出来。另外要注意，

```
\align ...\endalign
```

结构前面或后面却不许输入任何其它字符（除了空格）。

为了解决数学公式的局部对齐问题，TEX 定义了用在数学公式中的另一种对齐结构：

```
\aligned ...\endaligned
```

在该结构的前面或后面可以输入其它的数学符号。例如公式

$$\left\{\begin{aligned}\alpha &= f(z)\\ \beta &f(z^2)\\ \gamma &= f(z^3)\end{aligned}\right\} \qquad \left\{\begin{aligned}x &= \alpha^2-\beta\\ y &= 2\gamma\end{aligned}\right\}$$

在源文件中的输入为：

```
$$
\left\{\aligned
\alpha&=f(z)\\  \beta &f(z^2)\\
\gamma& =f(z^3)\endaligned\right\}
\qquad \left\{\aligned x&=\alpha^2
-\beta\\ y&=2\gamma\endaligned\right\}
$$
```

如果你在 \aligned 的后面键入 \tag ...， TEX 将把编号打印在所对齐的那些公式的左 (或右) 边沿的中间位置。

§2.5.2.6 vanilla 格式中单列公式的分行

许多长数学公式都是由一些小的数学式子组成，而这些小公式又是用等号 “ = ” 或其它一些类似的符号 ≤ 、 ≥ 、 ⊂ 等连接而成。当这类长公式需要分行时，人们习惯在那些像等号一样的连接符号的前面分行：

$$\begin{aligned}(a+b)(a-b) &= (a+b)a-(a+b)b\\ &= a^2+ab-ab-b^2\\ &= a^2-b^2.\end{aligned}$$

这时你可以用 \align ...\endalign 这样的结构来输入上面的这种类型的公式。上面的这个公式在源文件中的输入为：

```
$$\align
(a+b)(a-b)&=(a+b)a-(a+b)b\\
&=a^2+ab-ab-b^2\\
&=a^2-b^2.\endalign
$$
```

你时常还会遇到下面这样的单列公式：

$$\begin{aligned}(f\circ g)''' &= \bigl[f'''(g(x))g'(x)^3+2f''(g(x))g'(x)g''(x)\bigr]\\ &\quad+\bigl[f''(g(x))g'(x)g''(x)+f'(g(x))g'''(x)\bigr]\end{aligned}$$

它的输入为：

```
$$\align (f\circ g)'''      &=\bigl[f'''
(g(x))g'(x)^3+2f''(g(x))g'(x)g''(x)\bigr]\\
&\qquad+\bigl[f''(g(x))g'(x)g''(x)+
f'(g(x))g'''(x)\bigr]
\endalign$$
```

其中第二个 & 号后面的控制词 \qquad 的作用是使公式的第二行的加号 “ + ” 之前的第四个空格和上一行公式中的等号对齐。

通常当你使用 \align ...\endalign 为有编号的长单列公式分行时，应该把编号在第一个双反斜杠的前面输入；但是当你在源文件的开始键入了 \TagOnRight 时，则应该把公式的编号在 \endalign 的前面键入。

§2.5.2.7 vanilla 格式中数学公式里的矩阵与分段函数

矩阵是由圆括号 (或其它的括号，如方括号) 和排放在括号之间的一组数据组成：

$$A=\left(\matrix{x-\lambda & 1 & 0\cr 0 & x-\lambda & 1\cr 0 & 0 & x-\lambda}\right)$$

这个等式在源文件中是按如下格式输入的：

```
$$
A=\left(\matrix
          x-\lambda & 1 & 0\\
          0 & x-\lambda & 1\\
          0 & 0 & x-\lambda
        \endmatrix
  \right).
$$
```

其中 \matrix ...\endmatrix 表示采用矩阵格式，双反斜杠 \\ 是矩阵中一行结束的标志，而符号 & 则表示一项数据的结束。在矩阵格式中符号 & 的作用和结构 \aligned ...\endaligned 中的 & 作用类似，它们都是要使几行公式中的某些符号对齐. 在 \matrix ...\endmatrix 中的 & 使每一列的数据在其中心位置对齐，并且在两个相邻的列之间留下一 \quad(约两个字符宽) 的空隙。还应该说明的是上面输入中的 \left(和 \right) 可以用在 2.5.2.2 中介绍的其它可变尺寸的括号或者绝对值号等代替，例如

```
$$
A=\left|\matrix
          x-\lambda & 1 & 0\\
          0 & x-\lambda & 1\\
          0 & 0 & x-\lambda
        \endamtrix
  \right|.
$$
```

的打印输出为行列式：

$$A=\left|\matrix{x-\lambda & 1 & 0\cr 0 & x-\lambda & 1\cr 0 & 0 & x-\lambda}\right|.$$

最常用的矩阵括号是圆括号，因此在 vanilla 格式中用 \pmatrix ...\endpmatrix 的结构去简化圆括号矩阵的输入。例如本段开始的那个矩阵也可以按下面的方法输入：

```
$$
A=\pmatrix
```

```
x-\lambda & 1 & 0\\
0 & x-\lambda & 1\\
0 & 0 & x-\lambda
\endpmatrix.
$$
```

\matrix...\endmatrix 和 \pmatrix...\pendmatrix 都可以被看作一个大数学符号，所以它也可以像其它数学符号一样在数学公式中使用。例如下面的矩阵公式

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0a+1b & 1a+0b \\ 0c+1d & 1c+0d \end{pmatrix} = \begin{pmatrix} b & a \\ d & c \end{pmatrix}$$

在源文件中的输入为：

```
$$\align
\pmatrix a&b\\c&d\endpmatrix
\pmatrix 0&1\\1&0\endpmatrix
&=\pmatrix 0a+1b&1a+0b\\
0c+1d&1c+0d\endpmatrix
\\ \vspace{1\jot}
&=\pmatrix b&a\\d&c\endpmatrix
\endalign$$
```

其中 \vspace\{1\jot} 的作用是加大两行公式间的空隙. 如果没有它，第一行和第二行之间的空隙就会太小：

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0a+1b & 1a+0b \\ 0c+1d & 1c+0d \end{pmatrix} = \begin{pmatrix} b & a \\ d & c \end{pmatrix}$$

如下的数据组

$$\begin{array}{cc} x & .1 \quad 1 \\ x+y & .01 \quad 10 \\ x+y+zA & .001 \quad 100 \end{array}$$

中的三列数据对齐的方式各不相同，并且第一列和第二数据之间的空隙与第二列和第三列数据之间的空隙也不同。第一列是在数据中心对齐，第二列是在数据左边对齐，而第三行则是在数据右边对齐。这种打印效果是在 \matrix ...\endmatrix 中 (在控制词 \matrix 的后面) 添加一行 \format...\\ 格式的命令而达到的。上面那组数据是按如下的方式输入：

```
$$\matrix
\format \c &\quad \l &\qquad\r\\
x&.1&1\\
x+y&.01&10\\
x+y+zA&.001&100
\endmatrix$$
```

其中第二行中的 \c 表示第一列数据以中心位置对齐； & 后的 \quad 表示第一列与第二列之间的空隙为一 \quad ； \l 表示第二列数据以其左边沿对齐；第二个 & 后的 \qquad 表示第二列和第三列数据之间的空隙为一 \qquad (等于二 \quad) ； 在第一个双反斜杠之前的 \r 表示第三列数据在其右边对齐。读者要特别注意这个格式的活用。换言之，应该把 \matrix ...\endmatrix 理解为数据表的输入结构而不是仅限于矩阵的输入结构。

分段函数

$$f(x)=\begin{cases}(x-1)^2, & \text{for } x>1\\ 0, & \text{for } -1\le x\le 1\\ (x+1)^2, & \text{for } x<-1.\end{cases}$$

的结构和矩阵类似，它可以用 \matrix ...\endmatrix 的格式输入：

```
$$
f(x)=\left\{\matrix
\format \l & \quad \l \\
(x-1)^2,&\text{for } x>1\\
0,&\text{for }-1\le x\le1\\
(x+1)^2,&\text{for $x<-1$.}
\endmatrix\right.
$$
```

为了简化分段函数的输入， TEX 专门定义了 \cases ...\endcases 来做为分段函数的输入格式。上面的输入等价于

```
$$
f(x)=\cases
(x-1)^2,&\text{for } x>1\\
0,&\text{for }-1\le x\le1\\
(x+1)^2,&\text{for $x<-1$.}
\endcases
$$
```

使用 \cases ...\endcases 格式输入分段函数时应该注意每一行仅有一个 & ；还要注意上面输

入中的句号不能改放在 \endcases 的后面，否则打印输出变为：

$$f(x)=\begin{cases}(x-1)^2, & \text{for } x>1\\ 0, & \text{for } -1\le x\le 1\\ (x+1)^2, & \text{for } x<-1\end{cases}.$$

其中句号的位置不对。

在打印矩阵时常使用的省略号表示方法如下： \hdots 表示水平方向的省略号， \vdots 表示垂直方向的省略号， \ddots 表示主对角线方向的省略号。例如，矩阵

$$\begin{pmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}$$

在源文件中的输入为：

```
$$\pmatrix
a_{11}&a_{12}&\hdots&a_{1n}\\
a_{21}&a_{22}&\hdots&a_{2n}\\
\vdots&\vdots&\ddots&\vdots\\
a_{n1}&a_{n2}&\hdots&a_{nn}
\endpmatrix
$$
```

练习 15. 在源文件中输入下面的数学公式：

$$\det\begin{pmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}=\begin{cases}1 & \text{if } a_{ij}=\delta_{ij}\\ (-1)^n & \text{if } a_{ij}=-\delta_{ij}\end{cases}$$

§2.5.2.8 vanilla 格式中数学公式里的省略号

我们知道在 vanilla 格式中， \dots 可以用在正文中表示省略号。而在数学公式中的省略号要复杂一些，例如公式

$$x_1+\cdots+x_n \qquad \text{and} \qquad (x_1,\ldots,x_n)$$

中的第一个省略号位于中间，而第二个省略号则位于底部。在数学公式中，可以分别用控制词 \cdots 和 \ldots 得到省略号 “ $\cdots$ ” 和 “ $\ldots$ ”，上面的数学公式在源文件中的输入为：

```
$$ x_1+\cdots+x_n\qquad
\text{and}\qquad (x_1,\ldots,x_n)
$$
```

在 vanilla 格式中的数学公式里，控制词 \ldots 和 \dots 都表示位于底部的省略号 “ $\ldots$ ”。通常， \cdots 用在加号 “ $+$ ”、减号 “ $-$ ”、等号 “ $=$ ” 和不等号 “ $\neq$ ” 的后面，而 \ldots 则用在逗号的后面。如果数学公式中省略号的后面紧跟着的是句号或是像) 和] 之类的符号，则应该在省略号的后面加一个小空隙 \,。例如，输入

```
So $A=1+x+x^2+\cdots \,$.
```

的输出是

So $A = 1 + x + x^2 + \cdots$.

又如

```
$$(1+x+x^2+\cdots\,)
(1+y+y^2+\cdots\,)$$
```

的打印结果为：

$$(1 + x + x^2 + \cdots)(1 + y + y^2 + \cdots)$$

§2.5.2.9 vanilla 格式中数学公式里的重音符号和字体

我们知道 TEX 把数学公式中的字母用 italic 体打印，但有时在数学公式中还会用到其它字体，例如在公式

$$a\mathbf{x} + b\mathbf{y}$$

中的 x 和 y 就是用 bold 体打印的。在 vanilla 格式的数学公式中， x 可以用 \bold x 来表示，所以上面的数学公式在源文件中的输入为：

```
$$a\bold x+b\bold y$$
```

注意控制词 \bold 是带参量的控制词，它和只能用在正文中的控制词 \bf 的用法不同。 \bf 不能在数学公式中使用，它不是带参量的控制词，其输入格式为： {\bf ...}；而 \bold 不能在正文中 (即数学公式之外) 使用，它是带参量的控制词，其输入格式为： \bold{...}。

在 vanilla 格式中的数学公式里还可以打印大写的 calligraph 字母 $\mathcal{A}$， $\mathcal{B}$， ...， $\mathcal{Z}$。这些字母在数学公式中分别用 \Cal A， \Cal B， ...， \Cal Z 表示。注意控制词 \Cal 中的字母 C 为大写字母 (不是小写字母)；还要注意 \Cal 仅能用在大写的英文字母之前。

在数学公式中，我们还经常用某些加上特殊重音标记的字母标识特定的意思。在 vanilla 格式里，数学公式中的表示这些重音标记的控制词和在正文中相应的那些 (表示重音标记的) 控制词的定义不同，下面例子中的控制词仅能在数学公式中使用：

```
$\hat a$      â
$\check a$    ǎ
$\tilde a$    ã
$\acute a$    á
$\grave a$    à
$\dot a$      ȧ
$\ddot a$     ä
$\breve a$    ă
$\bar a$      ā
$\vec a$      a⃗
```

如果你在数学公式中使用了控制符 \^ 或者是在正文中使用了控制词 \hat， TEX 就会在排版查错时给出相应的错误信息。

注意在数学公式中 \bar a 和 \overline a 的打印效果不同：

```
$\bar a$        ā
$\overline a$   a‾
```

而 \vec a 和 \overrightarrow a 的打印效果也不同：

```
$\vec a$              a⃗
$\overrightarrow a$   a⃗
```

在数学公式中也可以把重音符号加在其它字体的字母上面，例如

```
$\dot{\bold x}$   ẋ
$\hat{\Cal A}$    Â
```

事实上，你可以在任何数学式子的上面添加重音符，但是当数学式子比一个字母长时重音符号就会打印在式子上面的中间位置，这使得打印出来的公式很难看。为了解决这个问题， TEX 在数学公式中定义了一些可变尺寸的重音符号，例如

```
$\widehat x$       x̂
$\widehat{xyz}$    x̂yẑ
$\widetilde x$     x̃
$\widetilde{xy}$   x̃ỹ
```

§2.5.2.10 TEX 中数学公式里的小空隙

我们已经介绍过控制符 \，在数学公式中表示一个小空隙，应该在像 $f(x)\,dx$ 这样的数学公

式中使用。如果在数学公式中的阶乘符号 ! 的后面是其它的字母或者数字或者括号，也应该在 ! 的后面使用控制符 \, ：

```
$n!/\bigl(n!\,(n+1)\bigr)$
```
$n!/\bigl(n!\,(n+1)\bigr)$

```
$$\frac{4!}{3!\,2!}$$
```
$$\frac{4!}{3!\,2!}$$

同样，在根式和其它字符之间也应该有一个小空隙：

```
$\sqrt2\,x$
```
$\sqrt2\,x$

```
$(1+\sqrt n\,)^3$
```
$(1+\sqrt n\,)^3$

还有一些在数学公式中表示空隙的控制符，它们的用法都和 \, 相同，下面把这些控制符按所表示空隙大小的次序列举如下：

```
\,  \;  \quad  \qquad
```
\, \; \quad \qquad

有时在数学公式中我们还需要减少两个字符之间的空隙，可以用控制符 \! 来解决。下面给出了一些增加或减少空隙的例子：

```
$\sqrt{\,\log x}$
```
$\sqrt{\,\log x}$

```
$[\,0,1)$
```
$[\,0,1)$

```
$\log n\,(\log\log n)^2$
```
$\log n\,(\log\log n)^2$

```
$x^2\!/2$
```
$x^2\!/2$

```
$n/\!\log n$
```
$n/\!\log n$

```
$\Gamma_{\!2}+\Delta^{\!2}$
```
$\Gamma_{\!2}+\Delta^{\!2}$

```
$R_i{}^j{}_{\!kl}$
```
$R_i{}^j{}_{\!kl}$

```
$$\int_1^b\!\int_a^b$$
```
$$\int_1^b\!\int_a^b$$

§2.5.3 vanilla 格式的练习答案

这一段给出本节 15 个练习的答案。

练习 1.

```
 We have $|\alpha(x,y) - \beta(x,y)| \le
|10\alpha - 1| < |\beta|$, for all $\alpha \in A$,
$\beta\in B$ and $\gamma\in\Gamma$. But if
$\gamma\notin\Gamma$, then $f(\alpha,\beta) \ge
\{\gamma\}\ne-\infty$.
```

练习 2.

```
We have $$|\alpha(x,y) - \beta(x,y)| \le
|10\alpha - 1| < |\beta|,$$ for all $\alpha \in A$,
$\beta\in B$ and $\gamma\in\Gamma$. But if
$\gamma\notin\Gamma$, then $$f(\alpha,\beta) \ge
```

```
\{\gamma\}\ne-\infty.$$
```

练习 3.

```
If $a_1\le5$, $a_2\le 6$ and $\alpha\le7$, then
$$ (a_1 + a_2)^{2\alpha} \le 11^{14} = (11^7)^2.$$
```

练习 4.

```
$2^{2^{2^{2^{2^{2^{2^2}}}}}}$
```

练习 5.

```
$$f^{(n+2)}=[f^{(n+1)}]'=[f^{(n)}]''$$
```

练习 6.

```
$$\frac {\dfrac x2+1}2$$
```

练习 7.

```
$$\frac{\dbinom nk}2$$
$$\binom n{k/2}$$
$$\binom n{\frac 12k}$$
```

练习 8.

```
$$
0=(1+-1)^n
 =\sum^n_{j=0} (-1)^j \binom nj
$$
```

和

```
$$
\sum \Sb 1\le i\le p\\ 1\le j\le q\\
         1\le k\le r\endSb a_{ij}b_{jk}c_{ki}
$$
```

练习 9.

```
$$\int f(x) \, dx =
\int f(x)\frac{du}{dx}$$
```

和

```
$$\iiint\limits_M d\omega
= \iint\limits_{\partial M}\omega$$
```

练习 10.

```
$$ Ax^2+bx+c=a\left(x^2+\frac
bax+\frac ca\right)=a\left[\left
(x+\frac b{2a}\right)^2+\left(\frac
ca-\frac {b^2}{4a}\right)\right]
$$
```

练习 11.

```
$$m^\star\left(A\cap\biggl[\,
\bigcap_{i=1}^nE_i\biggr]\right)
=\sum_{i=1}^nm^\star(A\cap E_i)
```

```
$$
```

练习 12.

```
$$\frac{\max_{1<n<m}\log_2P_2}
{\lim_{n\to0}\frac{\sin x}x}$$
```

练习 13.

```
$$\sum_{1\text{ odd}}\binom n1=2^{n-1}
\qquad \text{for $n\ge1$}.$$
```

练习 14.

```
$$
\align
\alpha_4&=\sqrt{\frac12}\\
\alpha_8&=\sqrt{\frac12+\frac12\sqrt{\frac12}}\\
\alpha_{16}&=\sqrt{\frac12+\frac12\sqrt{\frac12
+\frac12\sqrt{\frac12}}}
\endalign
$$
```

练习 15.

```
$$
\align
\det&\pmatrix
a_{11}&a_{12}&\hdots&a_{1n}\\
a_{21}&a_{22}&\hdots&a_{2n}\\
\vdots&\vdots&\ddots&\vdots\\
a_{n1}&a_{n2}&\hdots&a_{nn}
\endpmatrix\\ \vspace{1\jot}
&\qquad=
\cases
1&\text{if $a_{ij}=\delta_{ij}$}\\
(-1)^n&\text{if $a_{ij}=-\delta_{ij}$}
\endcases\endalign
$$
```

§2.6 TEX 中打印信件的 ltr 格式

TEX 提供了用于打印信件的 ltr 格式. ltr 格式是在 vanilla 格式基础上修改而成的, 它只不过是添加了几条用来打印信头和信尾的控制词. 下面分别介绍 ltr 格式所特有的那些控制词的用法.

如果你想要使用 ltr 格式, 就必须在源文件的第一行键入

```
\input ltr.sty
```

以表示你采用 ltr 格式.

现在你可以输入回信地址。回信地址是在控制词 \returnaddress 和 \date 之间输入的，TEX 把它安排在版面的右上角打印。例如下一段输入

```
\returnaddress
Beijing Normal University
Beijing,100875,China.
\date{\today}
```

的输出为：

Beijing Normal University
Beijing,100875,China.

December 1, 2000

输入 \date{...} 中的省略号表示的是你写信的时间，\today 表示的是计算机内时钟的时间。

接下来应该输入收信人的地址。收信人的地址在控制词 \address 的下一行开始输入，例如我们可以输入

```
\address
Mr.~So and So,
His street and Number
His City, State
```

它所对应的输出是：

Mr. So and So,
His street and Number
His City, State

信的内容在控制词 \letter 的下一行开始输入，TEX 以齐头格式打印信的内容。写信人的姓名是在 \closing 的下一行输入，TEX 把它打印在版面的右下部。例如：

```
\closing
Sincerely Yours
Your Name
```

的输出是：

Sincerely Yours

Your Name

如果你还有一个简短的附言，可以在控制词 \annotations 后面输入。最后要以控制词 \endletter 作为信的结尾。

下面这一段源文件是用 ltr 格式输入的短信：

```
\input ltr.sty
\returnaddress

Beijing Normal University
```

```
Beijing,100875,China.
\date\today
\address
Mr.~So and So,
His street and Number
His City, State
\letter
Dear Mr.~So and So:

  I take great pleasure to recommend \dots to your
department, and I hope the information I give you
is helpful to your consideration.

Aren't you impressed with this typeset letter?
\dots

\closing
Sincerely yours
Your Name

\annotations
NY/ys
cc: What's his Name

\ps \TeX\ will allow you to type letters
like this too!
\endletter
\bye
```

这一段源文件所对应的打印输出结果见下一页。

Beijing Normal University
Beijing,100875,China.
December 1, 2000

Mr. So and So,
His street and Number
His City, State

Dear Mr. So and So:

I take great pleasure to recommend ... to your department, and I hope the information I give you is helpful to your consideration.

Aren't you impressed with this typeset letter? ...

Sincerely yours

Your Name

NY/ys
cc: What's his Name

P. S. TEX will allow you to type letters like this too!

§2.7 高级 TEX 命令

前面几节介绍了基本的 TEX 命令，应用这些命令可以得到令人满意的排版效果。在这一节里介绍一些高级 TEX 命令，以便我们更好地应用 TEX 。

§2.7.1 TEX 源文件的快速编译

人们在建立源文件时总免不了出错，因此运行 TEX 对它进行编译时会发现一些语法错误。如果想要找出源文件中的所有语法错误，就应该在 TEX 运行过程中用回车键来回答所有的错误信息。记录下 (通常是用打印机把 TEX 运行记录 .log 打印出来) 所有的错误类型和它在源文件中的行数，然后用文件编辑器修改源文件中的错误。

有时人们运行 TEX 的目的只是查找源文件中的错误。为了提高 TEX 查错的运行速度，可以在源文件中 \input vanilla.sty 的后面键入控制词 \syntax 。这时运行 TEX 对你的源文件查错，其速度要提高三到四倍。但是此时 TEX 只查找源文件中的语法错误而不进行排版，所以 TEX 运行的结果是只产生 .log 文件。当你纠正完源文件中的所有错误之后，就可以去掉源文件中的控制词 \syntax ，然后运行 TEX 对源文件排版产生 .dvi 文件。

§2.7.2 TEX 源文件的分解

当你要打印的文件很长时，最好把它们分解成几个独立的小源文件输入。可以对这些小源文件分别用 TEX 进行查错，分别纠正其中存在的错误，然后把这些小源文件转换成供打印机和屏幕显示用的大文件 .dvi ，这样可以节省查找源文件中的错误的时间。

下面说明如何把几个独立的源文件合并成一个大文件。首先我们建立源文件 try1.tex 如下：

```
\input vanilla.sty
This is the first paragraph. Nothing
fancy here, and it dosn't look
too good. but it will  look a lot
better when it's printed. To see why, just keep reading!

This is the second paragraph. First we'll
print the number  186,384.64.
(That's the
speed of light in miles/second.
I measured it in myself on 3/1/85).

\bye
```

建立源文件 try2.v 如下：

```
\input vanilla.sty
Hyphens are needed whenever we want     to get `compound´ words, like
``mother-in-law´´.  Both upper-
and lower-case letters can occur, as
```

```
in the word ``X-rated''.

You have to be {\it care\/}ful and {\it
thought\/}ful with {\it slanted\/} fonts.
You often need the ``{\it italic correc%
tion\/}'' for proper spacing, though not
before a {\it comma}, or a {\it period}.

\bye
```

确信这两个源文件中没有语法错误后，可以建立如下的名为 sum.tex(也可以用其它的文件名)的源文件把它们合并成一个大文件：

```
\input vanilla.sty
\input try1
\input try2.v
\bye
```

现在把 try1.tex 和 try2.v 中的第一行的 \input vanilla.sty 和最后一行的 \bye 去掉，然后对源文件 sum.tex 运行 TEX 排版（读者可以试试看不去掉 try1.tex 中的 \bye 会产生什么编译结果）. 这时 TEX 运行结束后生成的文件 sum.dvi 就是我们需要的大文件. 当你在屏幕上显示该文件时，计算机的屏幕上会出现如下的内容：

This is the first paragraph. Nothing fancy here, and it dosn't look too good. but it will look a lot better when it's printed. To see why, just keep reading!

This is the second paragraph. First we'll print the number 186,384.64. (That's the speed of light in miles/second. I measured it in myself on 3/1/85).

Hyphens are needed whenever we want to get 'compound' words, like "mother-in-law". Both upper- and lower-case letters can occur,as in the word "X-rated".

You have to be *care*ful and *thought*ful with *slanted* fonts. You often need the "*italic correction*" for proper spacing, though not before a *comma*, or a *period*.

TEX 中的控制词 \input 表示要读进一个外部文件，当要读入的外部文件的扩展名为 .tex 时，可以省略其扩展名. 如在源文件 sum.tex 中我们有：

```
\input try1
\input try2.v
```

另外，不要把 \input 后的文件名用括号括起来，即在源文件中不许有如下的输入：

```
\input{try1}
```

§2.7.3 TEX 源文件的自定义控制序列

当你要输入的源文件中有相同的一部分内容经常出现时，可以定义新的控制序列来表示这段内容. 这是 TEX 软件的突出优点，使我们能够大大地减少建立源文件的工作量. 例如，当文

章中的数学公式里经常出现式子 $\alpha^n+\beta^{n+1}$ 时，可以在源文件中第一次出现这个式子之前键入

```
\def\ab{\alpha^n+\beta^{n+1}}
```

现在就可以在源文件中用控制词 \ab 来表示 \alpha^n+\beta^{n+1}。例如，源文件中如下的一段输入

```
$\ab$ is a definition, $2^{\ab}$
$$f(n)=\frac1\ab$$
```

所对应的输出为：

$\alpha^n+\beta^{n+1}$ is a definition, $2^{\alpha^n+\beta^{n+1}}$

$$f(n)=\frac{1}{\alpha^n+\beta^{n+1}}$$

在源文件中定义新的控制序列的格式如下：

```
\def\...{...}
```

其中第一个省略号代表一个字母序列，第二个省略号为新定义的控制词所代表的内容。

新控制序列的定义最好放在源文件的开始部分，也可放在自然段的开始或者是在一个单列公式的前面，以便于查阅。如果在正文中定义新的控制序列，可能会使 TEX 在这个地方输出较大的空隙。也可以在单列公式中定义新的控制序列，不过这时新定义的控制序列仅在这一个公式内起作用，如果你在这个公式的外面用了这个控制序列，TEX 会给出错误信息说它无定义。

定义新控制序列要注意如下几方面的问题：

1. 一定要按规定的格式输入新定义的控制序列：\def 与反斜杠 \ 之间不能有空格，反斜杠 \ 与左花括号之间也不能有空格。
2. 不要在定义的花括号中键入没定义过的控制序列，也不许循环定义新的控制序列。源文件中不许有类似于下面的输入：

   ```
   \def\aaa{\aaa}
   ```

   也不许有类似于如下的循环定义结构存在：

   ```
   \def\vvvv{\uuuu}
   \def\uuuu{\vvvv}
   ```

3. 新定义的控制序列的名称不能和 TEX 中已经存在的控制序列的名称相同，否则会出现张冠李戴的现象。因此，在定义一串新控制序列后，最好用 TEX 试排版一次。如果新定义的控制序列名称和 TEX 已经定义过的控制序列名称相同，TEX 在排版时会通知你。

§2.7.4 TEX 源文件的自定义带有参量的控制词

为了简化源文件的输入，TEX 还提供了定义带参量的控制词的方法。如果在源文件中有许多如下形式的偏导数要输入：

$$\frac{\partial f}{\partial x}\quad\frac{\partial f}{\partial y}\quad\frac{\partial g}{\partial x}\quad\frac{\partial g}{\partial z}$$

你就可以先定义一个带有双参量的控制词 \pd 如下：

```
\def\pd#1#2{\frac{\partial #1}{\partial #2}}
```

(其中 \partial 在 TEX 中代表符号 ∂) 然后键入

```
$$\pd fx\qquad\pd fy\qquad\pd gx\qquad\pd gz$$
```

就可以得到上面的单列公式。实际上，这个输入中的 \pd fx 代表

```
\frac{\partial f}{\partial x}
```

同样 \pd fy 、 \pd gx 和 \pd gz 都表示类似的意思。

在 TEX 中我们可以定义带有多达 9 个参量的控制词，其定义格式如下：

```
\def\...#1#2#3...#n{...}
```

其中 \... 为所定义的控制词的名称； n 为 1 到 9 之间的一个自然数，它表示控制词带有参量的个数，称 #k($1 \le k \le n$) 为所定义控制词的第 k 个参量；而花括号中的省略号表示控制词所代表的内容。例如，下面的输入

```
\def\sm#1#2#3#4#5#6#7#8#9{
(#6+#7+#8+#1+#2+#3+#9)^{#1+#4+#5}}
```

定义了带有九个参量的控制词 \sm ，从而可用

```
$$\sm abcd5fghi$$
```

代替

```
$$(f+g+h+a+b+c+i)^{a+d+5}$$
```

来表示数学公式

$$(f+g+h+a+b+c+i)^{a+d+5}$$

定义带参量的控制词时要注意以下几点：

1. 在输入

   ```
   \def\... #1#2#3#4{...}
   ```

   中的 #1#2#3#4 必须是按数字的大小依次排列 (但是 #1 ， #2 ， ... ，却可以任何次序出现在花括号之中)，紧跟在 # 号后的数字不能超过 9 (即所带参量的个数不能超过九个) 。
2. 在第一个 # 号和左花括号之间不能有任何空格键入。
3. 当控制词所代表的内容中有参量作为上下标 (即 {...} 中有参量作为上下标) 时一定要把作为上下标的参量用花括号括起来。例如，你可以定义带参量的控制词 \vector 如下：

   ```
   \def\vector #1#2{(#1_1,\dots,#1_{#2})}
   ```

   这时输入 $$\vector x{n+1}$$ 对应的输出为：

   $$(x_1,\dots,x_{n+1})$$

   如果你在定义控制词时忘了把作为下标的 #2 用花括号括起来，即键入的是：

   ```
   \def\vector #1#2{(#1_1,\dots,#1_#2)}
   ```

   则输入 $$\vector x{n+1}$$ 对应的输出变为：

   $$(x_1,\dots,x_n+1)$$

   此时为了得到输出

   $$(x_1,\dots,x_{n+1})$$

   必须把输入改为

   ```
   $$\vector x{{n+1}}$$
   ```

§2.7.5 TEX 的分行和分页

在 §2.4.2 中我们介绍过控制词 \allowlinebreak 、 \linebreak 和 \newline 。还有另外一个控制分行的控制词 \nolinebreak ，它是禁止 TEX 分行的命令。如果键入

```
Don't stop here. \nollinebreak Continue
to say more on this line.
```

则 TEX 在排版时不会把分行点安排在句子 " Don't stop here. " 的后面。

在 §2.4.1 中，我们介绍过控制词 \smallpagebreak 、 \medpagebraeak 和 \bigpagebreak 的用法，它们都是只能用在自然段的后面增加两个自然段之间的距离。还有两个控制词 \pagebreak 和 \nopagebreak 也可以用来控制自然段之间的距离，但是它们所产生的输出效果却大不相同。\pagebreak 不仅可以用在自然段之间，它还可以在正文中的某一个单词的后边使用。当它出现在两个自然段之间时， TEX 把它后面的自然段安排在下一页来打印，同时尽量把它前面的自然段安排在本页的底部打印；当它出现在正文的某一单词的后面时， TEX 就先把这个单词所在的那一行尽量安排在本页的最底部打印出来，然后在新的一页打印下一行。\nopagebreak 的用法和 \pagebreak 类似，但二者表示的意思却相反。还有一个控制词 \newpage 也是用来分页的，TEX 看见它之后就另起一页打印其后的字符。

§2.7.6 TEX 中为插图等预留位置

有时文章中会有插图，这就需要 TEX 在排版时为插图留下适当的空间。如果你在源文件中的两个自然段之间键入带参量的控制词 \midspace{3in} ， TEX 在打印输出时就会在这两个自然段之间为插图多留下三英寸的空间 (当这两个自然段的分界处位于它们所在页的底部而没有足够的空间为插图留下时， TEX 就在下一页的顶部为插图留下空间) 。当插图还附有说明时，可把说明做为带参量的控制词 \caption 的参量。为插图留空同时又打印插图说明的输入格式如下：

```
\midspace{...} \caption{...}
```

其中第一个省略号代表数字和长度单位，第二个省略号代表插图说明。

如果要把插图放在本页的顶部，可用控制词 \topspace 代替 \midspace 。

需要强调的是：控制词 \caption 只能用在 \midspace 或 \topspace 的后面。

§2.7.7 vanilla 格式中默认格式的修改命令

在 TEX 的 vanilla 格式中有一些修改打印输出格式的命令，本节介绍一些有关这方面的控制词。

§2.7.7.1 vanilla 格式中单列数学公式编号的位置

在第 39 页的 §2.5.2.4 子节中提到过控制词 \tag... ，它的作用是表示单列公式的编号。在默认的情况下， vanilla 格式把单列数学公式的编号安放在公式左边，如果你想把所有单列公式的编号放在公式的右边，可以在你源文件开始的 \input vanilla.sty 之后添加一行

```
\TagsOnRight
```

§2.7.7.2 vanilla 格式中正文版面宽和高的调整

vanilla 格式默认纸的宽度为 8.5 英寸，高度为 11 英寸，它把文件打印输出 (不包括页眉和页码) 的宽度安排为 6.5 英寸，高度为 9 英寸，使得打印纸的四个边都留有 1 英寸的空白。

在 vanilla 格式中，可以通过带参量的控制词 \pagewidth 和 \pageheight 用来改变默认的正文版面的宽度和高度。这两个控制词的参量都是由数字和长度单位 (长度单位可以是英寸 in 、厘米 cm 、毫米 mm 以及 pt 和 pe) 组成。\pagewidth 的参量表正文版面的宽度，而 \pageheight 的参量表示正文编面的高度。

例如要打印出宽度为 6.25 英寸的文件，可以在位于源文件第一行

```
\input vanilla.sty
```

的下面添加如下的一行：

```
\pagewidth{6.25 in}
```

如果还想要把正文版面的高度改成 7 英寸，可在其后再添加一行：

```
\pageheight{7 in}
```

§2.7.7.3 vanilla 格式中正文版面位置的调整

在默认的情况下， vanilla 格式把正文版面的位置安放在打印纸张的中心位置，但是我们可以用带参量的控制词 \hcorrection 和 \vcorrection 改变正文版面在打印纸上打印的排放位置。这两个控制词的参量都是由数字和长度单位 (长度单位可以是英寸 in 、厘米 cm 、毫米 mm 以及 pt 和 pe) 组成，并且这里的数字可以是负数。 \hcorrection 的参量的绝对值是正文版面的水平平移量。若 \hcorrection 的参量为正，则表示把正文版面向右平移；如果为负，则表示把正文版面向左平移。\vcorrection 的参量的绝对值是正文版面的垂直移动量。若 \vcorrection 的参量为正，则表示把正文版面向下平移；如果为负，则表示把正文版面向上平移。

例如要想正文版面的位置往左移 $1/8 = 0.125$ 英寸，可以在位于源文件第一行

```
\input vanilla.sty
```

的下面添加如下的一行：

```
\hcorrection{-0.125 in}
```

如果还想把正文版面的位置向下移 $1/2 = 0.5$ 英寸，可再在添加一行

```
\vcorrection{0.5in}
```

§2.7.7.4 vanilla 格式中正文行距的调整

此外， vanilla 格式的默认正文行距为 12pt 。如果需要隔行打印 (有些杂志要求投稿预印本采用此格式)，则需要加大各行之间的距离。例如，要把行距增大到 18pt ，可以在源文件的开头，紧接着 \pagewidth{...} 和 \pageheight{...} 之后键入

```
\baselineskip18pt
```

在 TEX 中， \baselineskip 表示行距。

另一个有用的命令是

```
\midspace{...}
```

其中 ... 代表长度。该控制词可以用来增加两个自然段之间的距离。

§2.7.7.5 vanilla 格式中的页码

在默认的情况下，文章的页码总是从第一页开始编排。如果想使文章的页码从任意一个自然数开始，可以使用控制词

```
\pageno=n
```

其中 n 为一个自然数。例如，想使文章第一页的页码为 5，可以在源文件的开始部分添加一行

```
\pageno=5
```

如无需打印出文章的页码，可以在源文件的开头部分输入控制词：

```
\nopagenumbers
```

§2.7.7.6 vanilla 格式中不规则右边界的设置

有的人喜欢把文件的边缘打印成不规则的形状，这时可以使用控制词 \raggedright ，这个控制词的用法和 \it 的用法相同。例如，输入

```
{\raggedright Instead of producing pages with
justified lines. \TeX\ can produce page with
``ragged'' margins. You might prefer ragged margins for
a personal communication, since some people think that
justified margins are too formal, possibly giving the
impression that they were produce by an impersonal
computer.}
```

对应的输出为：

Instead of producing pages with justified lines. TEX can produce page with "ragged" margins. You might prefer ragged margins for a personal communication, since some
people think that justified margins are too formal, possibly giving the impression that they were produce by an impersonal computer.

§2.7.7.7 vanilla 格式中稍窄于正文的段落

为了突出某一段文字，人们常常把要突出的内容用稍微窄于正文的宽度打印出来。这时可以使用如下的输入格式：

```
\smallpagebreak
{\narrower ...   \par}
\smallpagebreak
```

其中省略号代表要突出的那一段内容。本书中的例子就是用这种方式输入的。

§2.7.7.8 vanilla 格式中的条目列表

在 TEX 中，带参量的控制词 \item 和 \itemitem 也可以产生特殊的输出效果，在这里以例子说明它们的用法。下面一段源文件

```
This is the first paragraph. Nothing
facy here, and it dosn't look
```

```
too good. but it will  look a lot

\item{1.}    better when it's printed.
To see why, just keep reading!
\itemitem{(a)}     This is the second
paragraph. First we'll
 print the number  186,384.64.
\itemitem{(b)}    That's the
speed of light in miles/second.
I measured it in myself on 3/1/85.
\item{2.} Hyphens are needed whenever we want
to get 'compound' words,
 like ''mother-in-law''.
```

对应的输出为：

This is the first paragraph. Nothing facy here, and it dosn't look too good. but it will look a lot

1. better when it's printed. To see why, just keep reading!
   (a) This is the second paragraph. First we'll print the number 186,384.64.
   (b) That's the speed of light in miles/second. I measured it in myself on 3/1/85.
2. Hyphens are needed whenever we want to get 'compound' words, like "mother-in-law".

§2.7.8 plain 格式正文版面的整体放大

在 plain 格式中，`\magnification` 代表排版的整体放大倍数，它的赋值方法如下：

```
\magnification=...
```

其中 ... 为一个数字，该数字应该等于放大倍数乘以 1000 。为了方便，TEX 定义了一些控制词表示放大倍数，现列举如下：

`\magstep0`	1 倍
`\magstephalf`	1.095 倍
`\magstep1`	1.2 倍
`\magstep2`	1.44 倍
`\magstep3`	1.728 倍
`\magstep4`	2.074 倍
`\magstep5`	2.488 倍

例如 TEX 源文件的第一行为

```
\magnification=\magstep1
```

则它表示把全文的排版整体放大 1.2 倍。

要注意的是本小节中的控制词在 vanilla 格式和 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中还有效。

§2.8 TEX 中的表和表格

TEX 提供两种打印表和表格的方法。第一种方法是模仿打字机的“制表键”，适用于可以延伸好几页的大表；第二种方法是用命令 \settabs 来制表。

§2.8.1 TEX 中用行对齐法编排表和表格

请看下面一段源文件：

```
$$\vbox {\tt \halign {\hfil # \hfil && \quad \hfil # \hfil\cr
$\backslash$rm & $\backslash$sl & $\backslash$it &
$\backslash$tt & $\backslash$bf \cr
\rm Roman & \sl Slanted & \it Italic & \tt Typewriter &
\bf Boldface\cr
}}$$
```

它所对应的打印输出为：

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

在本例的源文件中，第一个双美元符号 $$ 后面的带参量的控制词 \vbox 表示其后面的参量不是数学公式，因此 TEX 把参量中出现的字符按照正文的方式来处理，所以可在正文中使用的控制词都可以作为参量的一部分。另外，\vbox 还防止 TEX 把其参量分成两页来打印。因为在上面的表中我们要打印的大部分字符是打字机体的，所以 \vbox 的参量用控制词 \tt 作为开头。随后是带参量的控制词 \halign，它表示其参量要按照某种方式对齐。\halign 的参量是由若干语句组成的，每一个语句的结束是用控制词 \cr 表示。

\halign{...} 的参量中的第一个语句叫做序行 (preamble)，它规定表的列数及不同的列的对齐方式。序行的一般表达形式为：

```
...#...&...#...&&...#...\cr
```

下面介绍其中符号的含义：

- # 代表列，它在序行中的个数等于表的列数，除非在这里使用了 &&（将在后面解释该符号的含义）；
- & 代表列间隙，它在序行中的个数等于表的列数减 1，除非在这里使用了 &&（将在后面解释该符号的含义）；
- && 表示重复使用出现在 && 和 \cr 之间的列处理方式，并且这时表的列数等于组成各语句中的 & 的数加 1。

在序行结构中，... 由表示空隙的命令和 \hfil 构成，用于控制各个列的对齐方式，它是和表示列的符号 # 一起决定相应列的对齐方式，具体规则如下：

\hfil# 表示该列右对齐

#\hfil 表示该列左对齐

\hfil#\hfil 表示该列中心对齐

在序列行结构中，从 && 到 \cr 之间部分是可选项，它主要起到简化输入的作用。需要注意，当序列行中不包括 && 时，符号 # 和 & 必须交替出现，并且以 # 结束；当序列行中包括 && 时，在符号 && 之前符号 # 和 & 必须交替出现，并且以 # 结束。

现在回到前面的例子，这里 \halign 参量中的序行稍有不同，下面分别介绍这一行中各项的作用。 \hfil # \hfile 规定表的第一列的各项是在中心位置对齐；其后的两个 && 表示重复使用出现在 && 和 \cr 之间的处理方式； \quad 表示各列之间要添加一个 \quad 的额外空隙。

下面的表的三个列采取了不同的对齐方式：

1	2	3
The first column	The second column	The third column

它在源文件中的表示为：

```
$$\vbox {\halign{\hfil # \quad \quad &\hfil #\hfil \quad & #\hfil\cr
1&2&    3 \cr
The first column&The second column&The third column\cr
}}$$
```

由此读者可以体会表格中列的三种不同对齐方式。

有时我们事先并不知道应该在表的各列之间留多大的空隙（即 & 表示的空隙）才合适，这可以让 TEX 自己去决定。为此需要告诉 TEX 一个表示空隙大小的尺寸范围。考察如下一段输出：

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

它对应的源文件是：

```
$$\vbox {\tt \tabskip 1em plus 2em minus .5em
\halign to \hsize{\hfil # \hfil &&  \hfil # \hfil\cr
$\backslash$rm & $\backslash$sl & $\backslash$it &
$\backslash$tt & $\backslash$bf \cr
\rm Roman & \sl Slanted & \it Italic & \tt Typewriter &
\bf Boldface\cr
}} $$
```

此处

```
\tabskip 1em plus 2em minus .5em
```

表示各列的间隔是可以在 0.5em 到 3em 之间变化；

```
\halign to ...
```

表示尽量使表的总宽度达到 ... 的宽度；

```
\hsize
```

表示正文版面的宽度。

有时表中会有某一栏的内容太多，从而需在这一栏中打印两行文字，这时要用到带参量的控制词 \vtop。例如：

```
$$\vbox {\tt \tabskip 1em plus 2em minus .5em
\halign to \hsize{\hfil # \hfil & \hfil # \hfil &
\hfil # \hfil & \hfil # \hfil &
\vtop{\hsize=10em # }\cr
$\backslash$rm & $\backslash$sl & $\backslash$it &
$\backslash$tt &\hfil $\backslash$bf\hfil \cr
\rm Roman & \sl Slanted & \it Italic & \tt Typewriter &
{这是用黑体字打印出来的:\bf Boldface}\cr }} $$
```

在这一段源文件中，\vtop 的参量规定了表的最后一列的打印方式，其中的 \hsize=10em 定义了最后一列的宽度为 10em（这里的数字可以根据实际情况选取），当输入字符的宽度超过 10em 时 TEX 会自动在本栏中另起一行打印。这个例子所对应的输出为：

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	这是用黑体字打印出来的：**Boldface**

TEX 还可以打印出如下形式的表格：

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	这是用黑体字打印出来的：**Boldface**

这张表格和前面介绍的表之间的差别是各个栏目之间用直线隔开，这是名符其实的表格。这张表格在源文件中是按照如下的方式输入的：

```
$$\vbox {\tt \offinterlineskip
\halign to \hsize
{\tabskip=0pt \strut#& \vrule#\tabskip=1em plus2em &\hfil #\hfil &
\vrule#& \hfil # \hfil& \vrule#&\hfil#\hfil & \vrule#&\hfil #\hfil &
\vrule#& \vtop{\hsize=10em\strut # \strut}&\vrule# \tabskip=0pt\cr
\noalign{\hrule}& & $\backslash$rm & & $\backslash$sl & &
$\backslash$it & & $\backslash$tt & &\hfil $\backslash$bf\hfil& \cr
\noalign{\hrule}
& &\rm Roman & & \sl Slanted & & \it Italic & & \tt Typewriter & &
{这是用黑体字打印出来的:\bf Boldface} &\cr \noalign{\hrule} }} $$
```

下面简单介绍一下在这段源文件中出现的各项的含义。\tabskip=0pt 表示第一列前的距离为 0pt，它使得表格最左边的竖线作为各个表格横线的头；\offinterlineskip 表示各个行之间的距离为 0；\halign to \hsize 表示想要把表格的总宽度设制成正文的宽度。现在要把表格中的表示格子的每一个竖直线都看作一个不同的列；再往后是一个子体结构，这个子体是由若干

语句组成，每条语句都是以控制词 \cr 结尾（除了最后一条语句 \noalign{\hrule} ）。TEX 并不把序行的内容打印出来，该行规定了表格的打印格式。在序行中， \strut 定义了表格中每一行格的高度（这个高度随着每一行的字符的大小而变化）； \vrule 表示高度等于行高度的竖线； & \vrule# 是表格中的竖直线； \tabskip=1em plus2em 表示从第二列开使各个列之间的空隙可以在 1em 到 3em 之间变化（尽量取接近 1em 的量）；再往后的 \vrule#& 都是代表表格格子中的竖直线； tabskip=0pt\cr 表示作为表格最后一列竖直线的后面不留空隙。从第二语句开始至倒数第二句，每一个语句都依次表示表格中的一行，每一语句开头的 \noalign{\hrule} 代表该行上面的那一条表格横线，而最后一条语句后的 \noalign{\hrule} 代表表格最下边的直线。

下面的表格更为复杂：

张明的年龄、身高和体重		
年龄	身高	体重
1	56	6
2	70	10
3	90	19
4*		
5	120	30

*四岁时无记录

它在源文件中的输入为：

```
$$\vbox{\tabskip=0pt \offinterlineskip \halign to 200pt
{\strut#& \vrule#\tabskip=1em plus2em &\hfil ## & \vrule#&\hfil#
\hfil& \vrule#&\hfil#& \vrule#\tabskip=0pt\cr \noalign{\hrule}
& & \multispan5\hfil
    张明的年龄、身高和体重 \hfil&\cr \noalign{\hrule}
& & \omit\hidewidth 年龄 \hidewidth & &
\omit\hidewidth 身高 \hidewidth & &
\omit\hidewidth 体重 \hidewidth &\cr \noalign{\hrule}
& & 1 & & 56 & & 6 & \cr \noalign{\hrule}
& & 2 & & 70 & & 10& \cr \noalign{\hrule}
& & 3 & & 90 & & 19& \cr \noalign{\hrule}
& & 4\rlap* & & \multispan3 & \cr \noalign{\hrule}
& & 5 & & 120 & & 30& \cr \noalign{\hrule}
\noalign{\smallskip}
\multispan7*{ 四岁时无记录 } \hfil\cr}}
$$
```

其中 \multispan5 表示表格中第一行里的文字“张明的年龄、身高和体重”应该占 5 列的宽度（注意这里表格的竖线也算作列）； \omit 表示它的后一项不受序行规定的格式限制，它只能用在 \cr 或者 & & 之后； \hidewith...\hidewith 表示不以“ ... ”的宽度决定列的宽

度；在第七语句中的 \rlap 表示它后面的星号 * 在计算列的宽度时不起作用，而 \multispan3 使得输出表格中第六行出现空格； \noalign{\smallskip} 表示在竖直方向留出 \smallskip 的空隙；最后一条语句中的 \multispan7 表示 ''* 四岁时无记录'' 占七列的宽度，从而使得打印输出表格的最后一列中的竖直线消失；又因为表示最后一条语句结束的 \cr 的后面没有 \noalign{\hrule} ，所以 TEX 就不在表格的最后一列的下面划横线。

§2.8.2 TEX 中用固定列宽法编排表和表格

固定列宽法打印表和表格要用到控制词 \settabs ， 它打印出的同一列的各个项都是向左对齐的。下面的表就是用这种方法打印出来的：

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

它所对应的输入为：

```
$$\vbox {\tt \settabs 5 \columns
\+ $\backslash$rm & $\backslash$sl & $\backslash$it &
$\backslash$tt & $\backslash$bf \cr
\+\rm Roman & \sl Slanted & \it Italic & \tt Typewriter &
\bf Boldface\cr
}$$
```

其中 \+ 表示语句的开始； \cr 表示语句的结束； \columns 和它前面的数字表示所要打印的表的列数。

下面这一段源文件可以打印两个表和一个表格，它们采用的都是固定列宽法：

```
$$\vbox{\tt \settabs 5 \columns
\+\rm & \sl & \it &\tt & \bf\cr
\+\rm Roman & \sl Slanted & \it Italic &
\tt Typewriter & \bf Boldface\cr
}$$

$$\vbox{\tt \settabs 6 \columns
\+\hfill\rm \hfill&\hfill \sl \hfill&\hfill \it
\hfill&\hfill \tt \hfill&\hfill \bf\hfill&\cr
\+\hfill\rm Roman \hfill&\hfill \sl Slanted \hfill&\hfill
\it Italic \hfill&\hfill \tt Typewriter \hfill&
\hfill \bf Boldface\hfill&\cr
}$$

$$\vbox{\tt \settabs 8 \columns{\hrule}
\+\strut\vrule&\hfill\rm \hfill&\strut\vrule\,\strut\vrule&
\hfill \sl \hfill&\hfill \it
\hfill&\hfill \tt \hfill&\hfill\strut\vrule&\cr{\hrule}
\+\strut\vrule&\hfill\rm Roman \hfill&\strut\vrule\,\strut\vrule&
```

```
\hfill \sl Slanted \hfill&\hfill\it Italic \hfill&
\hfill \tt Typewriter \hfill&\hfill\strut\vrule&\cr{\hrule}
}$$
```

下面给出这段源文件对应的输出，读者可以对照输出结果进一步理解输入中的各部分的作用。

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

§2.8.3 TEX 中用可变列宽法编排表和表格

可变列宽法打印表和表格与固定列宽法打印表和表格的输入方式相类似. 把固定列宽法中的 \columns 和在它前面的数字用一个样本语句来代替，就可以得到可变列宽法的输入格式. 我们应该选择表中的最长的一行作为样本语句，并且样本语句的各个项之间要加适当的空格. 样本语句只起规定表的格式的作用， TEX 并不把它打印出来. 下面的表

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

和表格

\rm	\sl	\it	\tt	\bf
Roman	*Slanted*	*Italic*	Typewriter	**Boldface**

就是用可变列宽法打印的。

它们在源文件中分别是按照

```
$$\vbox{\tt \settabs \+\rm Roman &\quad \sl Slanted &\quad\quad
\tt Typewriter &\quad \bf Boldface&\cr
\+\hfill\rm \hfill&\hfill \sl \hfill&\hfill \it
\hfill&\hfill \tt \hfill&\hfill \bf\hfill&\cr
\+\hfill\rm Roman \hfill&\hfill \sl Slanted \hfill&\hfill
\it Italic \hfill&\hfill\tt Typewrite
\hfill&\hfill\bf Boldface \hfill&\cr
}$$
```

和

```
$$\vbox{\tt \settabs \+\quad\rm Roman\quad &&\quad \sl Slanted &
\quad \it Italic &\quad\tt Typewriter &\quad \bf Boldface&\cr{\hrule}
\+\strut\vrule\hfill\rm \hfill&\strut\vrule\strut\vrule&
```

```
\hfill \sl \hfill&\hfill \it\hfill&\hfill
\tt \hfill&\hfill \bf\hfill&\hfill\strut\vrule\cr{\hrule}
\+\strut\vrule\hfill\rm Roman \hfill&\strut\vrule\strut\vrule&\hfill
\sl Slanted \hfill&\hfill
\it Italic \hfill&\hfill \tt Typewriter \hfill&
\hfill \bf Boldface\hfill&\hfill\strut\vrule\cr{\hrule}
}$$
```

来编排的。

§2.9 TEX 中常用数学符号的表示

数学符号 + − = < > | / ( ) [ ] 和 * 可以用键盘上相应的键直接输入；此外，可以用 \vert 代替键盘上的 | 键来输入 | ，用 \labrak 和 \rbrack 代替 [和] 键，用 \ast 代替 * 键。下面给出其它数学符号的表示方法，特别注意他们只能在数学公式中使用。

§2.9.1 小写希腊字母

下面列出的这些小写希腊字母的定义不但可以在 TEX 的 Plain 和 Vanilla 格式使用，还可以在 $\mathcal{AMS}$-TEX 和 LATEX 中使用。

α	\alpha	β	\beta	γ	\gamma	δ	delta
ϵ	\epsilon	ε	\varepsilon	ζ	\zeta	η	\eta
θ	\theta	ϑ	\vartheta	ι	\iota	κ	\kappa
λ	\lambda	μ	\mu	ν	\nu	ξ	\xi
π	\pi	ρ	\rho	ϖ	\varpi	ϱ	\varrho
σ	\sigma	ς	\varsigma	τ	\tau	υ	\upsilon
ϕ	\phi	φ	\varphi	χ	\chi	ψ	\psi
ω	\omega						

§2.9.2 大写希腊字母

下面列出的这些大写希腊字母的定义不但可以在 TEX 的 Plain 和 Vanilla 格式使用，还可以在 $\mathcal{AMS}$-TEX 和 LATEX 中使用。

Γ	\Gamma	Δ	\Delta	Θ	\Theta	Λ	\Lambda
Ξ	\Xi	Π	\Pi	Σ	\Sigma	Υ	\Upsilon
Φ	\Phi	Ψ	\Psi	Ω	\Omega		

§2.9.3 大写花体英文字母

在 $\mathcal{AMS}$-TEX 中，大写花体英文字母 $\mathcal{A}$ ， $\mathcal{B}$ ， ... ， $\mathcal{Z}$ 分别用 \Cal A ， \Cal B ， ... ， \Cal Z 表示。注意这些花写体英文字母只能在数学公式中使用，此外在 LATEX 中，这些花写体英文字母是用 \mathcal A ， \mathcal B ， ... ， \mathcal Z 表示的。

§2.9.4 各种运算符号

下面列出的这些运算符号的定义不但可以在 TEX 的 Plain 和 Vanilla 格式使用，还可以在 AMS-TEX 和 LATEX 中使用。

±	\pm	∩	\cap	∨	\vee
∓	\mp	∪	\cup	∧	\wedge,\land
\	\setminus	⊎	\uplus	⊕	\oplus
·	\cdot	⊓	\sqcap	⊖	\ominus
×	\times	⊔	\sqcup	⊗	\otimes
*	\ast	◁	\triangleleft	⊘	\oslash
⋆	\star	▷	\triangleright	⊙	\odot
◇	\diamond	≀	\wr	†	\dagger
∘	\circ	◯	\bigcirc	‡	\ddagger
•	\bullet	△	\bigtriangleup	⨿	\amalg
÷	\div	▽	\bigtriangledown	&	\and

§2.9.5 各种逻辑关系符号

下面列出的这些逻辑关系符号的定义不但可以在 TEX 的 Plain 和 Vanilla 格式使用，还可以在 AMS-TEX 和 LATEX 中使用。

≤	\leq,\le	≥	\geq, \ge	≡	\equiv
≺	\prec	≻	\succ	∼	\sim
⪯	\preceq	⪰	\succeq	≃	\simeq
≪	\ll	≫	\gg	≍	\asymp
⊂	\subset	⊃	\supset	≈	\approx
⊆	\subseteq	⊇	\supseteq	≅	\cong
⊑	\sqsubseteq	⊒	\sqsupseteq	⋈	\bowtie
∈	\in	∋	\ni, \owns	∝	\propto
⊢	\vdash	⊣	\dashv	⊨	\models
⌣	\smile	∣	\mid	≐	\doteq
⌢	\frown	∥	\parallel	⊥	\perp
≠	\neq, \ne	∉	\notin		

\mid 和 \parallel 分别与 | 和 \| 打印出的符号相同，但是 \mid 和 \parallel 在所打印的符号两边留更大的空隙。上面列出的这些符号的否定式（除了 ≠ 和 ∉）可以在相应的控制词前面加上 \not 来表示。例如， \not\subset 表示 ⊄ .

§2.9.6 各种常用符号

ℵ	\aleph	′	\prime	∀	\forall
ℏ	\hbar	∅	\emptyset	∃	\exists
ı	\imath	∇	\nabla	¬	\neg, \lnot

$\jmath$	\jmath	$\surd$	\surd	$\flat$	\flat
ℓ	\ell	$\top$	\top	$\natural$	\natural
$\wp$	\wp	$\bot$	\bot	$\sharp$	\sharp
$\Re$	\Re	$\|$	|, \Vert	$\clubsuit$	\clubsuit
$\Im$	\Im	$\angle$	\angle	$\diamondsuit$	\diamondsuit
∂	\partial	$\triangle$	\triangle	$\heartsuit$	\heartsuit
∞	\infty	\	\backslash	$\spadesuit$	\spadesuit
$\int$	\smallint				

\prime 主要是用来作为上下标；用 \angle 作为上下标时其尺寸不变（除非你按照第 100 页中第 §3.9 节中介绍的方法从新定义它）.

§2.9.7 各种箭头

$\leftarrow$	\leftarrow, \gets	$\longleftarrow$	\longleftarrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow
$\rightarrow$	\rightarrow, \to	$\longrightarrow$	\longrightarrow
$\Rightarrow$	\Rightarrow	$\Longrightarrow$	\Longrightarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow
$\Leftrightarrow$	\Leftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\uparrow$	\uparrow	$\Uparrow$	\Uparrow
$\downarrow$	\downarrow	$\Downarrow$	\Downarrow
$\updownarrow$	\updownarrow	$\Updownarrow$	\Updownarrow
$\nearrow$	\nearrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\nwarrow$	\nwarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow
$\leftharpoonup$	\leftharpoonup	$\leftharpoondown$	\leftharpoondown
$\rightharpoonup$	\rightharpoonup	$\rightharpoondown$	\rightharpoondown
$\rightleftharpoons$	\rightleftharpoons		

附注：表示垂直箭头的控制词可跟在 \left 和 \right 的后面成为可变尺寸符号；\iff 也和 \Longleftrightarrow 一样表示 $\Longleftrightarrow$，只不过 \iff 在这个符号两边留的空隙稍微大一些。类似地，在 AMS-TEX 中还有两个控制词 \implies 和 \impliedby 分别表示符号 $\Longrightarrow$ 和 $\Longleftarrow$，它们自己两边留的空隙比控制词 \Longrightarrow 和 \Longleftarrow 留的空隙要稍微大一些.

§2.9.8 可变尺寸的数学符号

下面列出的符号的用法类似于求和号 $\sum$，这些符号都有两种尺寸. 当它们在正文中出现时 TEX 用小尺寸打印，当它们出现在单列公式中时 TEX 用大尺寸打印. 如果你想打印大尺寸符号，可以把 \dsize 加在相应的控制词之前；如果你想要打印小尺寸的符号，可以把 \tsize 加

在相应的控制词之前。

$\sum\sum$	\sum	$\bigcup\bigcup$	\bigcup	$\bigodot\bigodot$	\bigodot
$\prod\prod$	\prod	$\bigcap\bigcap$	\bigcap	$\bigotimes\bigotimes$	\bigotimes
$\coprod\coprod$	\coprod	$\bigsqcup\bigsqcup$	\bigsqcup	$\bigoplus\bigoplus$	\bigoplus
$\bigvee\bigvee$	\bigvee	$\biguplus\biguplus$	\biguplus	$\bigwedge\bigwedge$	\bigwedge

§2.9.9 各种积分号

$\int$	$\int$	\int	$\oint$	$\oint$	\oint
$\iint$	$\iint$	\iint	$\iiint$	$\iiint$	\iiint
$\iiiint$	$\iiiint$	\iiiint	$\int\cdots\int$	$\int\cdots\int$	\idotsint

上面这些符号都可以在 $\mathcal{AMS}$-TEX 中使用，它们也像求和号一样有两种尺寸，其尺寸选择方法也和求和号相同。注意，在 plain 格式中只有 \int 和 \oint 可以使用。

§2.9.10 各种括号

(	(	[	[, \lbrack	{	\{, \lbrace
)	)	]	], \rbrack	}	\}, \rbrace
$\lfloor$	\lfloor	$\lceil$	\lceil	$\langle$	\langle
$\rfloor$	\rfloor	$\rceil$	\rceil	$\rangle$	\rangle
|	|, \vert	$\|$	\\|, \Vert	/	/
\	\backslash				

表示各种上下箭头符号的控制词也可以像上面的表示各种括号的控制词一样跟在 \left 和 \right 的后面成为变尺寸符号。\left< 等价于 \left\langle，而 \right> 等价于 \right \rangle 。

§2.10 TEX 中的英文字体

在第 5 页中的 §1.2.4 小节中介绍了一些常用字体，这只是 TEX 字体的一小部分，本节中介绍其它字体的调用方法。

要使用 TEX 的一种字体，必须先定义一个控制词来表达该字体，然后就可以像 \rm 一样使用这个新定义的控制词了。定义表示行字体的方式如下：

```
\font...=*** scaled ~~~
```

其中 ... 是表示新字体的控制词；*** 是字库文件的名，~~~ 是字体的放大倍数乘以 1000 。可省略其中的 scaled ~~~ ，省略它表示放大倍数为 1 。通常人们取 ~~~ 为 1000 、 1095 、 1200 、 1440 、 1728 、 2074 和 2488 。进一步，在 TEX 和 $\mathcal{AMS}$-TEX 中可以使用第 62 页中 §2.7.8 节定义的控制词表示这些数。

例如我们想要使用名为 cmcsc10 ，并且还要使用它放大 1.44 倍后的字体，可以按如下方式定义：

```
\font\scten=cmcsc10
\font\scfourteen=cmcsc10 scaled 1440
```

现在可以用控制词 \scten 和 \scfourteen 分别表示 cmcsc10 字库中的字体原样和把它放大 1.44 倍后的字体. 在此定义下, 源文件

```
This {\scten is a \scfourteen example of} cmcsc10.
```

所对应的输出为

This IS A EXAMPLE OF cmcsc10.

为了读者使用方便, 我们在下一节中给出常用英文字符表.

另外, 还有一些字库存放的是各种字符. TEX 还提供了几种直接调用字库中字符的方法, 下面简单介绍一下. 可以用控制词

```
\char*
```

来表示字库中第 * 号位置的字符, 这里 * 是一个不超过 127 的非负整数. 假设我们已经定义好了表示 cmcsc10 字体的控制词 \scten, 则 {\scten\char10\char20} 对应的打印输出为

Ω˘

下表给出了 cmcsc10 字库部分字符的一种表达方式:

\char0	\char10	\char20	\char30	\char40	\char50	\char60
Γ	Ω	˘	Œ	(	2	<
\char70	**\char80**	**\char90**	**\char100**	**\char110**	**\char120**	**\char127**
F	P	Z	D	N	X	¨

字库中特定位置的字符还可以用结构

```
\symbol{*}
```

来表达, 其中 * 是一个不超过 127 的非负整数. 若我们已经定义好了表示 cmcsc10 字体的控制词 \scten, 则 {\scten\symbol{10}\qquad\symbol{20}} 对应的打印输出为:

Ω ˘

§2.11 TEX 和 AMS-TEX 及 LATEX 2ε 中的常用英文字符表

本节中给出一些常用字符表, 在每一个表的前面一行都给出了对应表的字库文件名, 改名之后的圆括号中给出了对应字库的英文字体名. 可以按 72 页 §2.10 小节中的方法使用这些字体或字符. 由于 TEX 字库采用 8 位编码, 所以每个字库只能存放 128 个字符. 下面各个字库中的字符都是顺序列出的, 每一行列出 4 组字符, 每一组有 8 个字符.

cmcsc10 (caps and small caps) (* \tensmc)

```
ΓΔΘΛΞΠΣΥ ΦΨΩ↑↓'¡¿ IJ`´ˆ˜¯˘˚ ¸ßSSÆŒØÆŒØ
-!"#$%&' ()*+,-./ 01234567 89:;<=>?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]ˆ`
```

‘ABCDEFG HIJKLMNO PQRSTUVW XYZ−−˝˜¨

cmex10 (Math extension) (\tenex)



cmbx10 (bold extended) (\tenbf or \bf)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ‘´ˇ˘¯˚ ¸ßæœøÆŒØ
-!”#$%&’ ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]ˆ‘
‘abcdefg hijklmno pqrstuvw xyz−−˝˜¨

cmb10 (10 point bold)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ‘´ˇ˘¯˚ ¸ßæœøÆŒØ
-!”#$%&’ ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]ˆ‘
‘abcdefg hijklmno pqrstuvw xyz−−˝˜¨

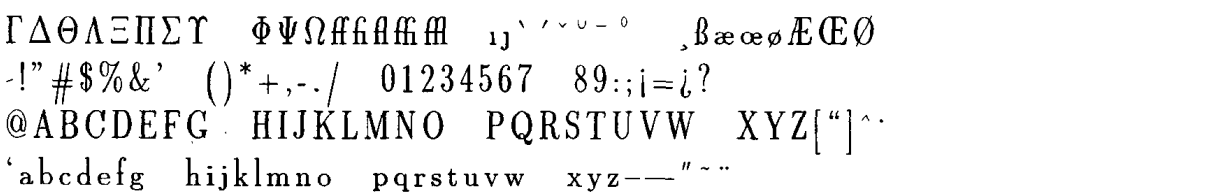
cmbxsl10 (bold extended slanted)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ‘´ˇ˘¯˚ ¸ßæœøÆŒØ
-!”#$%&’ ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]ˆ‘
‘abcdefg hijklmno pqrstuvw xyz−−˝˜¨

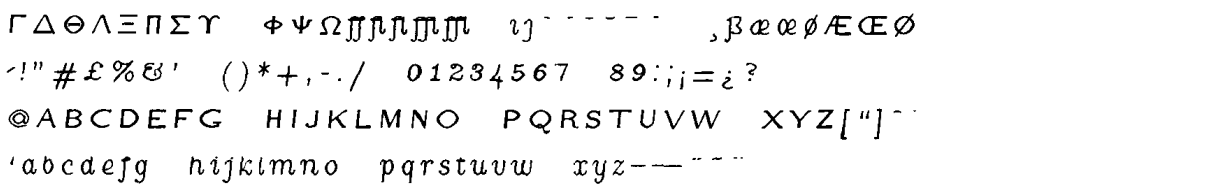
cmbxti10 (bold extended text italic)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ‘´ˇ˘¯˚ ¸ßæœøÆŒØ
-!”#£%&’ ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]ˆ‘
‘abcdefg hijklmno pqrstuvw xyz−−˝˜¨

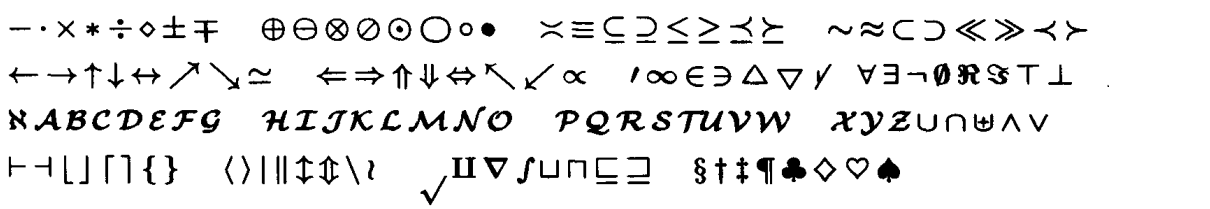
cmdunh10 (Dunhill)



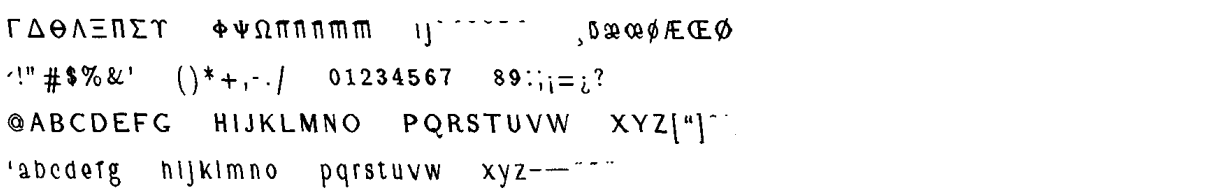
cmfi10 (funny italic)



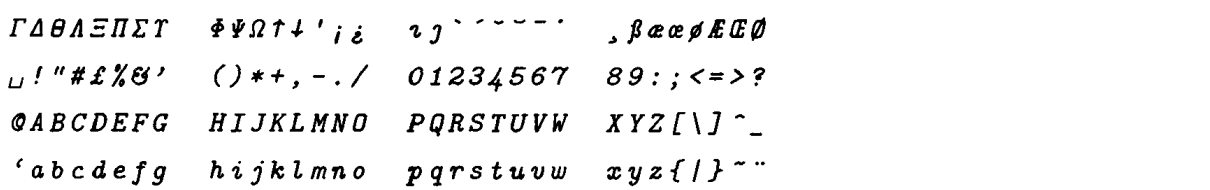
cmbsy10 (bold math symbol)



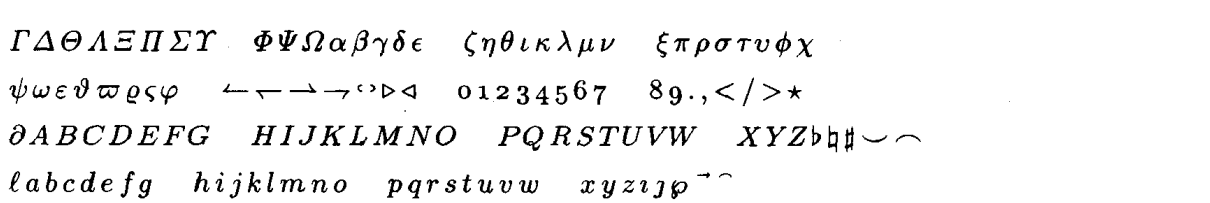
cmff10 (funny font)



cmitt10



cmmi10 (math italic) (\teni)



cmmib10 (math italic bold)

$\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Upsilon\ \Phi\Psi\Omega\alpha\beta\gamma\delta\epsilon\ \zeta\eta\theta\iota\kappa\lambda\mu\nu\ \xi\pi\rho\sigma\tau\upsilon\phi\chi$
$\psi\omega\varepsilon\vartheta\varpi\varrho\varsigma\varphi\ \leftharpoonup\leftharpoondown\rightharpoonup\rightharpoondown\triangleright\triangleleft\ 0123456 7\ 89.,</>\star$
$\partial ABCDEFG\ HIJKLMNO\ PQRSTUVW\ XYZ\flat\natural\sharp\smile\frown$
$\ell abcdefg\ hijklmno\ pqrstuvw\ xyz\imath\jmath\wp\vec{}\hat{}$

cmr10 (roman) (\rm or \tenrm)

ΓΔΘΛΞΠΣΥ ΦΨΩ↑↓'¡¿ ıȷ`´ˆ˜¯˘˙˚ ¸ßæœøÆŒØ
␣!"#$%&' ()*+,-./ 01234567 89:;<=>?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[\]^_
`abcdefg hijklmno pqrstuvw xyz{|}~¨

cmsl10 (slanted) (\tensl)

$\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Upsilon\ \Phi\Psi\Omega$ffififlffiffl ıȷ`´ˆ˜¯˘˚ ¸ßæœøÆŒØ
-!"#$%&' ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]^`
`abcdefg hijklmno pqrstuvw xyz–—˝˜¨

cmsltt10

$\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Upsilon\ \Phi\Psi\Omega\uparrow\downarrow{}'¡¿$ ıȷ`´ˆ˜¯˘˙ ¸ßæœøÆŒØ
␣!"#$%&' ()*+,-./ 01234567 89:;<=>?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[\]^_
`abcdefg hijklmno pqrstuvw xyz{|}~¨

cmss10 (san-serif)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ`´ˆ˜¯˘˚ ¸ßæœøÆŒØ
-!"#$%&' ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]^`
`abcdefg hijklmno pqrstuvw xyz–—˝˜¨

cmssbx10 (san-serif bold extended)

ΓΔΘΛΞΠΣΥ ΦΨΩffififlffiffl ıȷ`´ˆ˜¯˘˚ ¸ßæœøÆŒØ
-!"#$%&' ()*+,-./ 01234567 89:;¡=¿?
@ABCDEFG HIJKLMNO PQRSTUVW XYZ[“]^`
`abcdefg hijklmno pqrstuvw xyz–—˝˜¨

cmssdc10 (san-serif demibold condensed)

```
Γ Δ Θ Λ Ξ Π Σ Υ  Φ Ψ Ω ff fi fl ffi ffl  ı ȷ ` ´ ˘ ¯ ˚ ¸ ß æ œ ø Æ Œ Ø
-!"#$%&'  ()*+,-./  01234567  89:;¡=¿?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[“]^`
‘abcdefg  hijklmno  pqrstuvw  xyz––—˝˜¨
```

cmssi10 (san-serif italic)

```
Γ Δ Θ Λ Ξ Π Σ Υ  Φ Ψ Ω ff fi fl ffi ffl  ı ȷ ` ´ ˘ ¯ ˚ ¸ ß æ œ ø Æ Œ Ø
-!"#$%&'  ()*+,-./  01234567  89:;¡=¿?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[“]^`
‘abcdefg  hijklmno  pqrstuvw  xyz––—˝˜¨
```

cmsy10 (symbol) (\tensy)

```
−·×∗÷⋄±∓  ⊕⊖⊗⊘⊙◯∘•  ≍≡⊆⊇≤≥≼≽  ∼≈⊂⊃≪≫≺≻
←→↑↓↔↗↘≃  ⇐⇒⇑⇓⇔↖↙∝  ′∞∈∋△▽/  ∀∃¬∅ℜℑ⊤⊥
ℵABCDEFG  HIJKLMNO  PQRSTUVW  XYZ∪∩⊎∧∨
⊢⊣⌊⌋⌈⌉{}  ⟨⟩‖↕⇕\≀  √∐∇∫⊔⊓⊑⊒  §†‡¶♣♢♡♠
```

cmtcsc10

```
Γ Δ Θ Λ Ξ Π Σ Υ  Φ Ψ Ω ↑ ↓ ' ¡ ¿  ı ȷ ` ´ ˘ ¯ ˚ ¸ ss Æ Œ Ø Æ Œ Ø
␣!"#$%&'  ()*+,-./  01234567  89:;<=>?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[\]^_
‘ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ{|}˜¨
```

cmtex10

```
·↓αβ∧¬∈π  λγδ↑±⊕ϖ∂  ⊂⊃∩∪∀∃⊛↯  ←→≠⋄≤≥≡∨
 !"#$%&'  ()*+,-./  01234567  89:;<=>?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[\]^_
‘abcdefg  hijklmno  pqrstuvw  xyz{|}˜∫
```

cmti10 (text italic) (\tenit)

```
Γ Δ Θ Λ Ξ Π Σ Υ  Φ Ψ Ω ff fi fl ffi ffl  ı ȷ ` ´ ˘ ¯ ˚ ¸ ß æ œ ø Æ Œ Ø
-!"#£%&'  ()*+,-./  01234567  89:;¡=¿?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[“]^`
‘abcdefg  hijklmno  pqrstuvw  xyz––—˝˜¨
```

cmtt10 (typewriter type) (\tentt or \tt)

```
ΓΔΘΛΞΠΣΥ  ΦΨΩ↑↓'¡¿  ıȷ`´ˆ˜¯˘˙  ¸ßæœøÆŒØ
␣!"#$%&'  ()*+,-./  01234567  89:;<=>?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[\]^_
`abcdefg  hijklmno  pqrstuvw  xyz{|}~¨
```

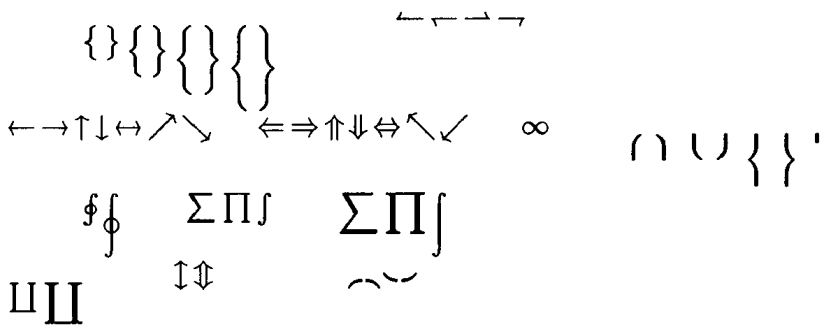
cmu10 (unslanted italic)

```
ΓΔΘΛΞΠΣΥ  ΦΨΩffififlffiffl  ıȷ`´ˆ˜¯˘˚  ¸ßæœøÆŒØ
-!"#£%&'  ()*+,-./  01234567  89:;¡=¿?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[“]^`
`abcdefg  hijklmno  pqrstuvw  xyz––”˜¨
```

cmvtt10

```
ΓΔΘΛΞΠΣΥ  ΦΨΩffififfiffl  ıȷ`´ˆ˜¯˘˙  ¸ßæœøÆŒØ
-!"#$%&'  ()*+,-./  01234567  89:;¡=¿?
@ABCDEFG  HIJKLMNO  PQRSTUVW  XYZ[“]^`
`abcdefg  hijklmno  pqrstuvw  xyz––˝˜¨
```

euex10



eufb10 (Euler Fraktur bold)

```
𝔞𝔣𝔣𝔤𝔱𝔱  𝔜3  ‘’
! &'  ()*+,−./  01234567  89:;=?
𝔄𝔅ℭ𝔇𝔈𝔉𝔊  ℌℑ𝔍𝔎𝔏𝔐ℕ𝔒  ℙℚℝ𝔖𝔗𝔘𝔙𝔚  𝔛𝔜ℨ[]^
𝔞𝔟𝔠𝔡𝔢𝔣𝔤  𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬  𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴  𝔵𝔶𝔷"1
```

eufm10 (Eular Fraktur medium) (* \teneufm)

```
𝔞𝔡𝔣𝔣𝔤𝔱𝔱𝔲  ‘’
! &'  ()*+,−./  01234567  89:;=?
```

ABCDEFG HIJKLMNO PQRSTUVW XYZ[]^
abcdefg hijklmno pqrstuvw xyz"1

eurb10 (Eular cursive bold)

ΓΔΘΛΞΠΣΥ ΦΨΩαβγδε ζηθικλμν ξπρστυφχ
ψωεϑϖ φ 01234567 89.,</>
∂ABCDEFG HIJKLMNO PQRSTUVW XYZ
ℓabcdefg hijklmno pqrstuvw xyzıȷ℘

eurm10 (Eular cursive medium)

ΓΔΘΛΞΠΣΥ ΦΨΩαβγδε ζηθικλμν ξπρστυφχ
ψωεϑϖ φ 01234567 89.,</>
∂ABCDEFG HIJKLMNO PQRSTUVW XYZ
ℓabcdefg hijklmno pqrstuvw xyzıȷ℘

eusb10 (Eular script bold)

— ~
¬ℜℑ
ℵABCDEFG HIJKLMNO PQRSTUVW XYZ ∧∨
{} | \ §

eusm10 (Eular script medium)

— ~
¬ℜℑ
ℵABCDEFG HIJKLMNO PQRSTUVW XYZ ∧∨
{} | \ §

lasy10 (extra symbols)

◁⊴▷⊵
‹›⌃⌄ ℧⋈□◇ ~⇝⊏⊐

lasyb10 (extra symbols bold version)

◁⊴▷⊵
‹›⌃⌄ ℧⋈□◇ ~⇝⊏⊐

logo10

A EF MNO PST

msam10 (computer modern bold math) (* \tenmsa)

msbm10 (computer modern bold math extra symbol) (* \tenmsb)

wncyb10 (cyrillic bold)

ЊЉЏЭІЄЂЋ њљџэієђћ ЮЖЙЁVѲSЯ южйёvѳsя
¨!”Ђ˘%´’ ()*Ћ,-./ 01234567 89:;«ı»?
ˇАБЦДЕФГ ХИЈКЛМНО ПЧРСТУВШ ШЫЗ[“]ЬЪ
‘абцдефг хијклмно пчрстувщ шыз––№ьъ

wncyi10 (cyrillic italic)

ЊЉЏЭІЄЂЋ њљџэієђћ ЮЖЙЁVѲSЯ южйёvѳsя
*¨!”Ђ˘%´’ ()*ћ,-./ 01234567 89:;«ı»?*
ˇАБЦДЕФГ ХИЈКЛМНО ПЧРСТУВШ ШЫЗ[“]ЬЪ
‘абцдефг хијклмно пчрстувщ шыз––№ьъ

wncyr10 (cyrillic upright)

ЊЉЏЭІЄЂЋ њљџэієђћ ЮЖЙЁVѲSЯ южйёvѳsя
¨!”Ђ˘%´’ ()*Ћ,-./ 01234567 89:;«ı»?

˘АБЦДЕФГ ХИЈКЛМНО ПЧРСТУВЩ ШЫЗ[“]ЬЪ
‘абцдефг хијклмно пчрстувщ шыз−—№ьъ

wncysc10 (cyrillic caps and small caps)

ЊЉЏЭІЄЂЋ њљџэієђћ ЮЖЙЁѴѲЅЯ южйёѵѳѕя
¨!”Ђ˜%´’ ()*ђ,-./ 01234567 89:;«й»?
˘АБЦДЕФГ ХИЈКЛМНО ПЧРСТУВЩ ШЫЗ[“]ЬЪ
‘АБЦДЕФГ ХИЈКЛМНО ПЧРСТУВЩ ШЫЗ−—№ЬЪ

wncyss10 (cyrillic upright san serif)

ЊЉЏЭІЄЂЋ њљџэієђћ ЮЖЙЁѴѲЅЯ южйёѵѳѕя
¨!”Ђ˜%´’ ()*ђ,-./ 01234567 89:;«ı»?
˘АБЦДЕФГ ХИЈКЛМНО ПЧРСТУВЩ ШЫЗ[“]ЬЪ
‘абцдефг хијклмно пчрстувщ шыз−—№ьъ

第三章 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 格式

用 TEX 所提供的格式文件可以打印信件和普通格式的文件，也可以打印常见的数学公式。虽然 TEX 提供了丰富的排版功能，但是要完成这些功能，需要极其复杂的 TEX 底层命令，这是一般的人所不能胜任的工作。为了简化排版工作， TEX 提供了创建格式文件的方法，使得 TEX 专家所编写的格式文件可以被一般的使用者所利用。目前世界上有很多的组织和 TEX 专家编制了适应于完成各类排版任务的格式文件，其中最著名的是美国计算机学家 Lesslie Lamport 开发的 LATEX 格式和美国数学协会开发的 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 格式。 LATEX 提供了一组生成复杂文档的高级 TEX 命令，使得一般会英语的使用者能够很容易地掌握这些命令，充分发挥 TEX 的强大功能。$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 提供了打印数学文章的高级 TEX 命令，其中包括各类数学符号的输入，数学文章的排版等。本章介绍 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 的使用方法。 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 程序包可以从下面的网址下载：

http://www.ams.org/tex/

§3.1 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 源文件的编译与特点

在 emtex 软件包（或其他的 TEX 软件包）所在的目录下，通常会有名为 amstex.bat 的批处理文件，该批处理文件专为运行 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 而编写。运行 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 编译源文件的命令格式与 §1.3.1 段中的运行 TEX 的命令格式类似：

```
amstex [d:][path]filename↩
```

其中 [d] 表示源文件所在的驱动器， [path] 表示源文件所在的路径， filename 表示源文件的全名。当源文件的扩展名为 .tex 时可以省略其扩展名；当源文件就位于当前的目录下可以省略方括号部分的输入。

运行 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 的结果会产生两个文件 .dvi 和 .log 。有了 .dvi 文件之后就可以按照 §1.1 节中的方法显示或打印输出排版结果。

$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 是在 plain 格式的基础上添了一些数学结构的定义而成，因此它具备了 plain 格式的基本特性。用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 可以打印 vanilla 格式文章，这只需在源文件的第一行键入

```
\input vanilla.sty
```

或者

```
\documentstyle{vanilla}
```

即可。

用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 还可以借助于 ltr 格式打印信件，这只需在源文件的第一行键入

```
\input ltr.sty
```

或者

```
\documentstyle{ltr}
```

即可。

$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 兼容于 TEX 所提供的原始格式文件，并且还提供了表示众多数学符号的简单命令的方法，因此在这里推荐使用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 打印数学文章。

§3.2 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中源文件的组成部分与 amsppt 格式

采用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX格式的源文件有如下的结构:

```
\documentstyle{...}
 { ... }
 { ... } 文章的版面定义部分
 { ... }
\topmatter
 { ... }
 { ... } 文章的标题部分
 { ... }
\endtopmatter
\document
 { ... }
 { ... } 文章的主体部分
 { ... }
\enddocument
```

在上面的结构中, \documentstyle{...} 是决定源文件所采用格式控制词; 在 \topmatter 和 \endtopmatter 之间的部分是文章标题部分的描述; 在 \document 和 \enddocument 之间的部分是文章的内容部分。

在 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中, 定义了一个新命令 \documentstyle{...} , 它的作用是表示源文件所采用的格式文件, 其中的省略号 ... 表示的是格式文件的名 (不包括扩展名 .sty) . 例如

```
\documentstyle{aaa}
```

表示采用格式文件 aaa.sty 所定义的格式.

当然可用命令 \input... 完成类似功能, 但对懂英语的人来说 \documentstyle{...} 更利于理解与记忆,因为 documentstyle 恰好是由英文单词 document(表示文件的意思)和 style (表示风格的意思) 所构成。 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 就是用这种方法来定义一些常用的排版命令, 极大地方便了使用者, 本节中介绍其中一些常用的命令与文件格式.

amsppt 格式是美国数学会规定的数学论文预印格式 (作为尚未在杂志上发表论文的初级版本), 该格式由格式文件 amsppt.sty 所定义。在这种格式中, 文章的标题、作者姓名、命题、证明以及参考文献等等, 都已经预先设计好了编排方式, 因而使用起来格外方便. 一般地讲, 这是一种最常用的数学论文打印格式。 \documentstyle{amsppt} 表示源文件采用 amsppt 格式。在接下来的几节中我们介绍 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX源文件的各个组成部分.

§3.3 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中 amsppt 格式的版面定义

采用 amsppt 格式的源文件以如下的方式开头：

```
\documentstyle{amsppt}
{ ...
  ...
  ... }  文章的版面定义部分
```

其中花括号部分表示一些确定排版定义的命令。当选用默认格式时，可以省略这一部分，直接进入文件标题的编排。

在 amsppt 格式中，常用的版面定义命令如下：

1. \def{...}：该命令用于定义控制序列，其具体的使用方法见 §2.7.3(第 56 页)和 §2.7.4（第 57 页）；
2. \pageno{...}：用来表示文章第一页的打印页码，例如 pageno=26 表示在打印输出的页码从 26 开始算起（即第一页的页码为 26，第二页的页码为 27，...）；
3. \pagewidth{...}：该命令规定了文章页面排版的宽度，例如 \pagewidth{6in} 表示页面排版的宽度为 6 英寸；
4. \pageheight{...}：该命令规定了文章页面排版的高度，例如 \pageheight{7in} 表示页面排版的高度为 7 英寸；
5. \NoPageNumbers：表示不把页码打印出来；
6. \NoRunningHeads：表示不打印页眉，此时如果打印页码，页码会出现在底部的中间位置；
7. \Monograph：表示打印的是书中的一章，它的使用方法在 §3.6 节（第 88 页）中介绍；
8. \loadbold：表示装入 bold 字体。

§3.4 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中 amsppt 格式的标题部分

在采用 amsppt 格式的源文件中，控制词 \topmatter 和 \endtopmatter 之间是输入文章的标题、作者、通信地址、关键词、摘要等的位置，本书称这部分的源文件内容为标题部分。对同一作者来说，每次打印新论文时，只需对标题部分略作修改。标题部分的结构如下：

```
\toppmatter
\title...\endtitle
\author...\endauthor
\affil...\endaffil
\address...\endaddress
\email...\endemail
\dedicatory...\enddedicatory
```

```
\date...\enddate
\thanks...\endthanks
\translator...\endtranslator
\keywords...\endkeywords
\subjclass...\endsubjclass
\abstract...\endabstract
\endtopmatter
```

下面分别介绍这些命令的用法。

1. `\title...\endtitle` ： 该结构用来输入文章的标题，其用法和 §2.3.1（第 19 页）段中完全相同。
2. `\author...\endauthor` ： 该结构用来输入文章的作者，其用法和 §2.3.1 （第 19 页）段中完全相同。
3. `\affil...\endaffil` ： 用来输入文章作者的单位，其中省略号代表作者的单位，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它打印在作者姓名下面一行的中间位置。
4. `\address...\endaddress` ： 用来输入文章作者通信地址的，其中省略号代表作者通信地址，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它打印在文章的参考文献之后。
5. `\email...\endemail` ： 用来输入作者的电子邮件地址，它需要和
   `\address...\endaddres`
   一起使用，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它作为文章的最后一行来打印。
6. `\dedicatory...\enddedicatory` ： 用来输入作者的献词，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它用 italic 字体打印在文章摘要的前面。
7. `\date...\enddate` ： 用来输入文章的完成日期，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它放在
   `\dedicatory...\enddedicatory`
   的前面。
8. `\thanks...\endthanks` ： 用来输入作者致谢的结构，它是作为无标号的脚注打印在文章首页的底部。如果文章的感谢词由几个自然段构成，只能用 `\endgraf`
   而不能用空行或 `\par` 来表示自然段的结束。
9. `\translator...\endtranslator` ： 用来输入翻译文章者的姓名（当文章是从非英文文字翻译成英文时就要有译者姓名），$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它打印在参考文献之后。
10. `\keywords...\endkeywords` ： 用来输入文章的关键词，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它作为没有标号的脚注打印在第一页的底部。
11. `\subjclass...\endsubjclass` ： 用来输入文章的分类号，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它作为没有标号的脚注打印在第一页的底部。
12. `\abstract...\endabstract` ： 用来输入摘要，$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把它用小一点的字体打印在正文的前面，并且摘要的排版宽度也比正文稍微窄一点。

特别注意，本节介绍的控制词只能用在 amsppt 格式中。还要注意在

\topmatter 和 \endtopmatter

之间的控制词是成对出现，不能忘记键入 \end... 。如键入了 \tilte 而忘记键入 \endtilte 或键入了 \abstract 而忘记键入 \endabstract 等， $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 在排版时给出的错误信息为：

```
! Paragraph ended befor \next@ was complete
```

在 \title...\endtitle 中可以使用 \\ 表示断行点输入多行标题，同样也可以在下面的三个结构中使用 \\ 表示断行点：

```
\author...\endauthor
\affil...\endaffil
\dedicatory...\enddedicatory
```

除了上面说明的这四个输入结构外，在 \topmatter 和 \endtopmatter 之间的其它输入结构中都不能使用 \\ 表示断行点，只能用控制词 \newline 来另起一行。

§3.5 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中 amsppt 格式的正文部分

在 amsppt 格式中，控制词 \document 和 \enddocument 之间是输入文章的正文位置，这里 \enddocument 表示源文件的结束(回想在 Plain 格式中和 Vanillla 格式中，总是用 \bye 表示源文件的结束)。正文中可以有标题、小标题、定义、定理、证明以及参考文献等等，在 amsppt 格式中它们都有固定的输入格式。下面分别介绍这些输入格式。

正文中的标题有如下四种水平的输入格式

```
\specialhead...\endspecialhead
\head...\endhead
\subhead...\endsubhead
\subsubhead...\endsubsubhead
```

其中 \specialhead...\endspecialhead 和 \head...\endhead 中可以用 \\ 表示断行点输入多行标题，而其它的标题输入格式中不能用 \\ 表示断行点。

定理和证明的输入已经在 §2.3.3 中介绍过，这里可以同样使用。在 amsppt 格式中还提供了如下三种输入格式：

```
\definition...\enddefinition
\example...\endexample
\remark...\endremark
```

它们分别用来输入定义、例子和注解，其使用方法与 §2.3.3 中的

```
\proclaim...\endproclaim
```

的使用方法相同。

有时还需要在正文中为插图留下合适的空间，这就要用到如下的输入格式：

```
\topinsert\quad 或者 \quad \midinsert
\captionwidth{ 长度 }  (为可选择项)
\topcaption{ 插图说明标题 }
 插图说明文字
\endcaption
```

```
\vspace{ 长度 }
\endinsert
```

其中 \topinsert 表示把插图的位置放在版面的顶部； \midinsert 表示把插图的位置放在版面的中间； \captionwideth 的长度表示插图说明的宽度（其长度单位可以是英寸 in 、厘米 cm 和 pt 等等）； \topcaption 表示把插图说明打印在插图的顶部，如果你想把插图说明打印在插图的底部可以用 \botcaption 来代替它； \vspace 的长度表示插图的高度.

在 amsppt 格式的源文件中，可以用如下的格式输入参考文献：

```
\Refs
\ref...\endref
\ref...\endref
...
\ref...\endref
\endRefs
```

其中 \ref...\endref 是一条参考文献. 下面简单介绍一下在输入参考文献时常用的控制词。

在 amsppt 格式中，每一条参考文献都可以用 \ref...\endref 这样的结构输入，并且还有一些专门用来输入参考文献的编号、作者姓名和文献题目等等的命令. 例如在

```
\ref\no10
\by V. I. Arnol$'$d
\paper The small dispersion limit for the
KDV equation
\jour Comm. Pure Appl. Math.
\vol 36 \yr 1983 \pages 123--145
\endref
```

中， \ref 和 \endref 分别表示一条参考文献的开始和结束； \no 的后面输入的是文献的编号（有时也可以用 \key 代替它输入文献的索引字符）； \by 的后面输入的是文献作者的姓名； \paper 表示它后面输入的是一篇论文的题目（这里也可以用 \book 表示后面输入的是一本书的名称）； \jour 表示其后输入的是发表论文的杂志名称； \vol 后的数字表示杂志的卷号； \yr 后的数字表示杂志的年份； \pages 后输入的是文献在杂志上的页码.

在用如上的结构输入参考文献时，要特别注意的是标点符号的输入，结构 \ref...\endref 中的任何控制词前面的那个字符不应该是标点符号（当然要除了表示缩写词的句号），$\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 在打印输出时会自动添加应该有的标点符号.

还要注意，用上述结构输入参考文献时，文献的编号不能超过两位数. 如果参考文献编号超过两位数，则需要用控制词 \widestnumber 通知 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX. 例如，当要输入的参考文献的最大编号为三位数字时，可在控制词 \Refs 的下一行键入

```
\widestnumber\no{999}
```

同样，当参考文献的索引字符超过两个字符时也需要通知 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX . 例如，当要输入的参考文献最长的索引字符为 GHMaR 时，需要在相应的地方添加如下一行：

```
\widestnumber\no{GHMaR}
```

如果我们按照 amsppt 格式所规定的格式输入参考文献，则可以在正文中用带参量的控制

词 \cite{...} 来表示参考文献的引用，例如 \cite{21} 的输出为 [21] 。

§3.6 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中用 amsppt 格式打印书

用 amsppt 格式可以把书以章为单位编排成不同的源文件，控制词 \Monograph 正是为此目的而定义。用 amsppt 格式打印书中各文章的源文件具有如下的输入结构：

```
\documentstyle{amsppt}
\Monograph
\topmatter
\title\chapter{...}...\endtitle
\endtopmatter

\document
...
\enddocument
```

其中 \chapter{...}... 用来输入章的编号和题目。例如

```
\documentstyle{amsppt}
\Monograph
\topmatter
\title\chapter{3}
This is the Title\endtitle
\endtopmatter
```

所对应的输出为：

CHAPTER III

THIS IS THE TITLE

注意输入中的 3 在输出时自动变为罗马数字 III ，并在数字的前面自动添加了词 CHAPTER 。如果你想把 CHAPTER III 换成其它的字符，可以使用控制词 \nofrills ：

```
\topmatter
\title\chapter\nofrills{APPENDIX B} This is the Title\endtitle
\endtopmatter
```

上面这一段源文件对应的打印结果为：

APPENDIX B

THIS IS THE TITLE

这时 APPENDIX B 代替了前面输出中的 CHAPTER III 。

如果你想打印的是书中的一篇前言，则可省略 \chapter 命令：

```
\topmatter
\title Preface\endtitle
```

书中的目录也可以作为一个独立的文章打印，这时需要用到控制词 \toc 和 \endtoc 。打印一本书的目录的源文件有类似于如下的结构：

```
\topmatter
\title Contents\endtitle
\endtopmatter
\bigskip
\document
\toc
\title Preface\page{vii}\endtitle
\title\chapter{1} Matrix Algebras
\page{1}\endtitle
\head Continuous complex-valued functions
\page{1}\endhead
...
\title Bibliography\page{307}\endtitle
\endtoc
\enddocument
```

其中 \toc...\endtoc 是表示打印目录；而 \title...\page{...}\endtitle 中的第一个省略号表示书中某一章节的标题，第二个省略号表示该章节的开始页码； \chapter{...}... 中的第一个省略号表示章的编号，第二个省略号表示章的标题，例如 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 把这里的

```
\chapter{1} Matrix Algebras
```

打印成 Chapter I. Matrix Algebras ；类似地， \head... 中的省略号表示节的标题.

§3.7 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中用 amsppt 格式打印数学论文的例子

本节给出一段源文件和相应的输出，以便于读者更好地理解前面几节介绍的控制序列．另一方面，读者也可以把它作为构造 amsppt 格式源文件的框架.

§3.7.1 amsppt 格式的源文件

下面是一个 amsppt 格式的源文件的全部内容，它所对应的打印结果在 §3.7.2 段中给出.

```
\magnification=\magstep1
\documentstyle{amsppt}
\pagewidth{27pc}
\pageheight{42pc}
\NoRunningHeads
\pageno=97
\def\nln{\newline}

\topmatter
\title The Title is Book
\endtitle
```

```
\author author \endauthor
\affil (Beijing Normal University) \endaffil
\address Department of Mathematics, Beijing Normal University,
Beijing 100875, The People's Republic of China \endaddress
\email name\@bnu.edu.cn\endemail
\dedicatory Dedicated to professor X on the occasion of his
eightieth birthday. \enddedicatory
\date December, 10, 1991
\enddate
\thanks The author was partially supported by the Ying-Tung
Fok Educational Foundation and the National Natural Science
Foundation of China
\endthanks
\translator lll. A
\endtranslator
\keywords ads, ghj   \endkeywords
\subjclass 60J25 \endsubjclass
\abstract Given a Dirichlet form on $L_2(R^d,qdx):$
$$\Cal E (f,g)=\int\,\sum_{i,j}\nabla_i fa_{i,j} \nabla_j g\,qdx\,, $$
where $a_{i,j}(x)$ is a nonnegative definite matric-valued
function of $x$. Let $(X_t)$ be the diffusion process associated to
$\Cal E$ (see [{\bf 1}] and [{\bf 2}] for details). When $a_{i,j}(x)$ and
$q(x)$ are smooth and strictly positive, $(X_t)$ is a semimartingale.
\endabstract
\endtopmatter

\document

\head  \S1. This is head   \endhead
Nevertheless, we will show that, by using the idea
from stochastic mechanics to consider the diffusion  process
forwardly and backwardly along the time axis and to find their
relationship, one can deal with these symmetric diffusion
processes much more easily. We will assume throughout this paper
that there is a $\kappa >0$ such that $\kappa^{-1} < a_{i,j}<\kappa$
and $\kappa^{-1} < q < \kappa$.
\subhead This is subhead \endsubhead
In the paper, we prove the following result:
\proclaim{Theorem 1.1} This is a paper.
\endproclaim
\demo{Proof} Obvious. \enddemo
```

```
That is  a man.
$$\Cal A\Cal B\Cal C\Cal D\Cal E\Cal F\Cal G\Cal H\Cal I\Cal J\Cal K
\Cal L\Cal M\Cal N\Cal O\Cal P\Cal Q\Cal R\Cal S\Cal T\Cal U\Cal V
\Cal W\Cal X\Cal Y\Cal Z
$$
\medpagebreak

It is O.K.

\medpagebreak

$\alpha \beta \chi $
$\delta \eta \varphi $
$\gamma \kappa \lambda $
$\mu \nu \omega $
$\pi \psi \rho $
$\sigma \tau \theta $
$\varepsilon \xi \zeta $
$\Lambda \Delta \Phi \Gamma \Omega \Pi \Sigma \Theta \Xi $


\medpagebreak

$\Bbb A \Bbb B$ $\Bbb C \Bbb D \Bbb E \Bbb F$ $\Bbb G \Bbb H \Bbb I
\Bbb J$ $\Bbb K \Bbb L \Bbb M \Bbb N$ $\Bbb O \Bbb P$ $\Bbb Q \Bbb R
\Bbb S$ $\Bbb T$ $\Bbb U \Bbb V \Bbb W \Bbb X$ $\Bbb Y \Bbb Z$

\medpagebreak

$\frak A\frak a$ $\frak B\frak b$ $\frak C \frak c$  $\frak D \frak d$
$\frak E \frak e$ $\frak F \frak f$ $\frak G \frak g$ $\frak H \frak h$
$\frak I \frak i$ $\frak J \frak j$ $\frak K \frak k$ $\frak L \frak l$
$\frak M \frak m$ $\frak N \frak n$ $\frak O \frak o$ $\frak P \frak p$
$\frak Q \frak q $ $\frak R \frak r $ $\frak S \frak s$ $\frak T
\frak t$ $\frak U \frak u $ $\frak V \frak v $ $\frak W \frak w $
$\frak X \frak x$ $\frak Y \frak y $ $\frak Z \frak z$

\medpagebreak

\P\ \$\ \&\ \#\ \dag\ \ddag\ \copyright\ \OE\ \oe\ \AE\ \ae\
\AA\ \aa\ \O\ \o\ \L\ \l\ \ss\
```

```
\ae uvre en fran\c cais

\medpagebreak

\`o \'o \^o \"o \~o \u o \v o \H o \B o \b o \D o \d o \c o \t oo

\medpagebreak

It is fine.

Final step. $2^{2^{2^{2^{2^{2^{2^2}}}}}}$  $2_{2_{2_{2_{2_{2_{2_2}}}}}}$

$$\frac 1{1+\frac 1{2+\frac 1{3+\frac 1{4+\frac 1{5+\frac1\ddots}}}}}
\qquad a_0+\cfrac1\\a_1+\cfrac1\\a_2+\cfrac1\\a_3+\cfrac1\\
\ddots\endcfrac$$
$$\pmatrix a_{11}&a_{12}&\cdots&a_{1n}\\
a_{21}&a_{22}&\cdots&a_{2n}\\
\vdots&\vdots&\ddots&\vdots\\
a_{n1}&a_{n2}&\cdots&a_{nn}
\endpmatrix\qquad \text{and}\qquad \binom n{n-k}\tag $\ast$
$$
$$ \root 5\of{x+y}+\sqrt{1+\sqrt{2+\sqrt{3+\sqrt{4+
\sqrt{5+\frac xy}}}}}\tag$\ast\ast$
$$
$$f(x,y)=\cases
x, &\text{for }1<y\le2\\
x+1, &\text{for }2<y\le3\\
x-1, &\text{for }3<y\le4\\
x^2 &\text{for }4<y\le5\\
x^2+1, &\text{for }5<y\le6\\
x^2-1, &\text{for }6<y\le7\\
0, &\text{otherwise}
\endcases
$$
$$\align
g_1(x,y)&=\sin x\cos x \log y \exp{(x+y)}\sum_{i=1}^n\alpha(i)\\
g_2(x,y)&=\sin x\cos x \log y \int_i^\infty h(x)\,dx
\endalign
$$
$$\overbrace{A+B+C+D+E+F+G+H+I+J+K+L+M+N}$$
```

```
$$\undersetbrace 14\to {A+B+C+D+E+F+G+H+I+J+K+L+M+N}$$

$$\CD G@>\alpha>>H\\
@VfVV@AgAA\\
G'@<<\beta<H'
\endCD
$$
Many classical methods can be applied to its study. But when
$a_{i,j}(x)$ and $q(x)$ are not smooth, the corresponding process
may be not a semimartingale. It creates some new difficulties in
the research.
\Refs

\ref\no 1\by Lyons T. J. and Zheng, W. A.\paper A Crossing Estimate
for the Canonical
Process on a Dirichlet Space and a
Tightness Result \jour Colloque
Paul Levy sur les Processus Stochastiques, Asterisque \yr
1988\pages 157-158, 249-271 \endref

\ref\no 2\by Lyons, T. J. and Zheng, W. A.
\paper Conditional Diffusion Processes\jour to
appear in the Proceeding of Edinburgh Mathematical Society
\yr 1989\endref

\widestnumber\key{GHMaR}

\ref\key C1
\by B. Coomes
\book Polynomial flows, symmetry groups, and conditions
 sufficient for injectivity of maps
\bookinfo Ph.D. thesis, Univ. Nebraska--\penalty0\hskip0pt Lincoln
\yr 1988
\endref

\ref\key C2
\bysame % B. Coomes
\paper The Lorenz system does not have a polynomial flow
\jour J. Differential Equations
\toappear
\endref
```

```
\ref\key GHMaR
\by J. Guckenheimer, P. Holmes, M. Martineau, and L. P. Robinson
\book Nonlinear oscillations, dynamical systems, and
 bifurcations of vector fields
\bookinfo
\publ Springer-Verlag \publaddr New York
\yr 1983
\endref

\endRefs

\enddocument
```

§3.7.2 源文件的打印效果

接下来的四页是 §3.7.1 段中给出的源文件的打印结果。

THE TITLE IS BOOK

AUTHOR

(Beijing Normal University)

December, 10, 1991

Dedicated to professor X on the occasion of his eightieth birthday.

ABSTRACT. Given a Dirichlet form on $L_2(R^d, qdx)$:

$$\mathcal{E}(f,g) = \int \sum_{i,j} \nabla_i f a_{i,j} \nabla_j g \, q dx \,,$$

where $a_{i,j}(x)$ is a nonnegative definite matric-valued function of x. Let (X_t) be the diffusion process associated to $\mathcal{E}$ (see [1] and [2] for details). When $a_{i,j}(x)$ and $q(x)$ are smooth and strictly positive, (X_t) is a semimartingale.

§1. THIS IS HEAD

Nevertheless, we will show that, by using the idea from stochastic mechanics to consider the diffusion process forwardly and backwardly along the time axis and to find their relationship, one can deal with these symmetric diffusion processes much more easily. We will assume throughout this paper that there is a $\kappa > 0$ such that $\kappa^{-1} < a_{i,j} < \kappa$ and $\kappa^{-1} < q < \kappa$.

This is subhead. In the paper, we prove the following result:

1991 *Mathematics Subject Classification.* 60J25.

Key words and phrases. ads, ghj.

The author was partially supported by the Ying-Tung Fok Educational Foundation and the National Natural Science Foundation of China

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

Theorem 1.1. *This is a paper.*

Proof. Obvious.

That is a man.

$$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

It is O.K.

$\alpha\beta\chi\ \delta\eta\varphi\ \gamma\kappa\lambda\ \mu\nu\omega\ \pi\psi\rho\ \sigma\tau\theta\ \varepsilon\xi\zeta\ \Lambda\Delta\Phi\Gamma\Omega\Pi\Sigma\Theta\Xi$

$\mathbb{AB}\ \mathbb{CDEF}\ \mathbb{GHIJ}\ \mathbb{KLMN}\ \mathbb{OP}\ \mathbb{QRS}\ \mathbb{T}\ \mathbb{UVWX}\ \mathbb{YZ}$

$\mathfrak{Aa}\ \mathfrak{Bb}\ \mathfrak{Cc}\ \mathfrak{Dd}\ \mathfrak{Ee}\ \mathfrak{Ff}\ \mathfrak{Gg}\ \mathfrak{Hh}\ \mathfrak{Ii}\ \mathfrak{Jj}\ \mathfrak{Kk}\ \mathfrak{Ll}\ \mathfrak{Mm}\ \mathfrak{Nn}\ \mathfrak{Oo}\ \mathfrak{Pp}\ \mathfrak{Qq}\ \mathfrak{Rr}\ \mathfrak{Ss}\ \mathfrak{Tt}\ \mathfrak{Uu}\ \mathfrak{Vv}\ \mathfrak{Ww}\ \mathfrak{Xx}\ \mathfrak{Yy}\ \mathfrak{Zz}$

¶ $ & # † ‡ © Œ œ Æ æ Å å Ø ø Ł ł ß
æuvre en français

ò ó ô ö õ ǒ ŏ ő ō o̲ ȯ ọ ǫ o͡o

It is fine.

Final step. $2^{2^{2^{2^{2^{2^{2^{2}}}}}}}$ $2_{2_{2_{2_{2_{2_{2_{2}}}}}}}$

$$\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{3+\cfrac{1}{4+\cfrac{1}{5+\cfrac{1}{\ddots}}}}}}$$

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{\ddots}}}}$$

(*)
$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \quad \text{and} \quad \binom{n}{n-k}$$

(**)
$$\sqrt[5]{x+y}+\sqrt{1+\sqrt{2+\sqrt{3+\sqrt{4+\sqrt{5+\frac{x}{y}}}}}}$$

$$f(x,y) = \begin{cases} x, & \text{for } 1 < y \le 2 \\ x+1, & \text{for } 2 < y \le 3 \\ x-1, & \text{for } 3 < y \le 4 \\ x^2 & \text{for } 4 < y \le 5 \\ x^2+1, & \text{for } 5 < y \le 6 \\ x^2-1, & \text{for } 6 < y \le 7 \\ 0, & \text{otherwise} \end{cases}$$

$$g_1(x,y) = \sin x \cos x \log y \exp(x+y) \sum_{i=1}^{n} \alpha(i)$$

$$g_2(x,y) = \sin x \cos x \log y \int_i^{\infty} h(x)\,dx$$

$$\overbrace{A+B+C+D+E+F+G+H+I+J+K+L+M+N}$$

$$\underbrace{A+B+C+D+E+F+G+H+I+J+K+L+M+N}_{14}$$

$$\begin{array}{ccc} G & \xrightarrow{\alpha} & H \\ {\scriptstyle f}\downarrow & & \uparrow{\scriptstyle g} \\ G' & \xleftarrow[\beta]{} & H' \end{array}$$

Many classical methods can be applied to its study. But when $a_{i,j}(x)$ and $q(x)$ are not smooth, the corresponding process may be not a semimartingale. It creates some new difficulties in the research.

References

1. Lyons T. J. and Zheng, W. A., *A Crossing Estimate for the Canonical Process on a Dirichlet Space and a Tightness Result*, Colloque Paul Levy sur les Processus Stochastiques, Asterisque (1988), 157-158, 249-271.
2. Lyons, T. J. and Zheng, W. A., *Conditional Diffusion Processes*, to appear in the Proceeding of Edinburgh Mathematical Society (1989).

C1. B. Coomes, *Polynomial flows, symmetry groups, and conditions sufficient for injectivity of maps*, Ph.D. thesis, Univ. Nebraska–Lincoln, 1988.

C2. ———, *The Lorenz system does not have a polynomial flow*, J. Differential Equations (to appear).

GHMaR. J. Guckenheimer, P. Holmes, M. Martineau, and L. P. Robinson, *Nonlinear oscillations, dynamical systems, and bifurcations of vector fields*, Springer-Verlag, New York, 1983.

Translated by LLL. A

Department of Mathematics, Beijing Normal University, Beijing 100875, The People's Republic of China
E-mail address: name@bnu.edu.cn

§3.8 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中定义的扩充字体

在本节中我们简单介绍一下 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中所扩充的字体，这些字体只能用在数学公式之中。在 amsppt 格式中可以直接使用这些新字体，在其它格式中则需要用另外一些控制词使得这些字符可以打印出来。下面我们总是假设使用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 进行排版。

在 amsppt 格式中， fraktur 字体可以用控制词 \frak 得到：

$\mathfrak{A}$	\frak A	$\mathfrak{B}$	\frak B	$\mathfrak{C}$	\frak C	$\mathfrak{D}$	\frak D
$\mathfrak{E}$	\frak E	$\mathfrak{F}$	\frak F	$\mathfrak{G}$	\frak G	$\mathfrak{H}$	\frak H
$\mathfrak{I}$	\frak I	$\mathfrak{J}$	\frak J	$\mathfrak{K}$	\frak K	$\mathfrak{L}$	\frak L
$\mathfrak{M}$	\frak M	$\mathfrak{N}$	\frak N	$\mathfrak{O}$	\frak O	$\mathfrak{P}$	\frak P
$\mathfrak{Q}$	\frak Q	$\mathfrak{R}$	\frak R	$\mathfrak{S}$	\frak S	$\mathfrak{T}$	\frak T
$\mathfrak{U}$	\frak U	$\mathfrak{V}$	\frak V	$\mathfrak{W}$	\frak W	$\mathfrak{X}$	\frak X
$\mathfrak{Y}$	\frak Y	$\mathfrak{Z}$	\frak Z				
$\mathfrak{a}$	\frak a	$\mathfrak{b}$	\frak b	$\mathfrak{c}$	\frak c	$\mathfrak{d}$	\frak d
$\mathfrak{e}$	\frak e	$\mathfrak{f}$	\frak f	$\mathfrak{g}$	\frak g	$\mathfrak{h}$	\frak h
$\mathfrak{i}$	\frak i	$\mathfrak{j}$	\frak j	$\mathfrak{k}$	\frak k	$\mathfrak{l}$	\frak l
$\mathfrak{m}$	\frak m	$\mathfrak{n}$	\frak n	$\mathfrak{o}$	\frak o	$\mathfrak{p}$	\frak p
$\mathfrak{q}$	\frak q	$\mathfrak{r}$	\frak r	$\mathfrak{s}$	\frak s	$\mathfrak{t}$	\frak t
$\mathfrak{u}$	\frak u	$\mathfrak{v}$	\frak v	$\mathfrak{w}$	\frak w	$\mathfrak{x}$	\frak x
$\mathfrak{y}$	\frak y	$\mathfrak{z}$	\frak z				

在 amsppt 格式中 blakboard bold 字体可以用控制词 \Bbb 得到：

$\mathbb{A}$	\Bbb A	$\mathbb{B}$	\Bbb B	$\mathbb{C}$	\Bbb C	$\mathbb{D}$	\Bbb D
$\mathbb{E}$	\Bbb E	$\mathbb{F}$	\Bbb F	$\mathbb{G}$	\Bbb G	$\mathbb{H}$	\Bbb H
$\mathbb{I}$	\Bbb I	$\mathbb{J}$	\Bbb J	$\mathbb{K}$	\Bbb K	$\mathbb{L}$	\Bbb L
$\mathbb{M}$	\Bbb M	$\mathbb{N}$	\Bbb N	$\mathbb{O}$	\Bbb O	$\mathbb{P}$	\Bbb P
$\mathbb{Q}$	\Bbb Q	$\mathbb{R}$	\Bbb R	$\mathbb{S}$	\Bbb S	$\mathbb{T}$	\Bbb T
$\mathbb{U}$	\Bbb U	$\mathbb{V}$	\Bbb V	$\mathbb{W}$	\Bbb W	$\mathbb{X}$	\Bbb X
$\mathbb{Y}$	\Bbb Y	$\mathbb{Z}$	\Bbb Z				

如果你使用的不是 amsppt 格式，就需要在控制词 \documentstyle{...} 的下面添加如下几行：

```
\loadmsbm
\loadeufm
```

才能使用上面的两种字体。

§3.9 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中定义的数学符号

在 amsppt 格式的数学公式中可以打印如下的符号：

✓	\checkmark	Ⓡ	\circledR	✠	\maltese	¥	\yen
⌜	\ulcorner	⌝	\urcorner	⌞	\llcorner	⌟	\lrcorner
⇢	\dashrightarrow	⇠	\dashleftarrow				

如果你使用的不是 amsppt 格式，就需要在控制词 \documentstyle{...} 的下面添加如下一行：

```
\UseAMSsymbols
```

表示你要使用 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中增加的数学符号。在源文件的开头键入 \UseAMSsymbols 后，就可以使用上面列出的控制词去打印相应的符号。

如果计算机的内存比较小，就不能装入所有 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中定义的数学符号，从而也就不能使用控制词 \UseAMSsymbols 。这时可以使用 \newsymbol 来定义控制词表示你想要打印的数学符号。首先需要在 \documentstyle{...} 的下一行键入

```
\loadmsam
\loadmsbm
\loadeufm
```

然后定义控制词：

```
\newsymbol...
```

其中 ... 是你所定义的控制词和它所表达的数学符号代码（数学符号的代码可以从后面列出的表中查到）。例如，数学符号 $\nleqslant$ 对应的代码为 230A ，可以在源文件中用如下的方法定义它：

```
\newsymbol\nleqslant 230A
```

有了这个定义后，就可以用控制词 \nleqslant 表示数学符号 $\nleqslant$ 。当然你也可以选用其它的控制词来定义此符号：

```
\newsymbol\nnnn 230A
```

这样就可以用 \nnnn 来表示符号 $\nleqslant$ 。

我们还可以给已有定义的控制词赋予新的含义。例如 \angle 代表 $\angle$ ，但是在 vanilla 格式中用它作上下标时其尺寸不变，而在 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX 中用它作为上下标时其尺寸却会自动变小，使打印的效果更好。所以当你要用 $\angle$ 作为上下标时就需要重新定义控制词 \angle（如果你用的是 amsppt 格式或者在源文件的开头已经键入了 \UseAMSsymbols ，就不需要重新定义它了）：

```
\undefine\angle
\newsymbol\angle 105c
```

下面列出了一个数学符号表，它给出了数学符号、相应的代码及其在 amsppt 格式中的定义，在其后标有 ''(U)'' 的符号在使用时需要重新定义（除非你用的是 amsppt 格式或者键入了命令 \UseAMSsymbols）。

小写希腊字母

$\digamma$	207A	\digamma	$\varkappa$	207B	\varkappa

希伯来字母

$\beth$	2069	\beth	$\gimel$	206A	\gimel
$\daleth$	206B	\daleth			

常用符号

$\hbar$	207E	\hbar (U)	$\backprime$	1038	\backprime
$\hslash$	207D	\hslash	$\varnothing$	203E	\varnothing
$\vartriangle$	134D	\vartriangle	$\blacktriangle$	104E	\blacktriangle
$\triangledown$	104F	\triangledown	$\blacktriangledown$	1048	\blacktriangledown
$\square$	1003	\square	$\blacksquare$	1004	\blacksquare
$\lozenge$	1006	\lozenge	$\blacklozenge$	1007	\blacklozenge
$\circledS$	1073	\circledS	$\bigstar$	1046	\bigstar
$\angle$	105C	\angle (U)	$\sphericalangle$	105E	\sphericalangle
$\measuredangle$	105D	\measuredangle			
$\nexists$	2040	\nexists	$\complement$	197B	\complement
$\mho$	2066	\mho	$\eth$	2067	\eth
$\Finv$	2060	\Finv	$\diagup$	231E	\diagup
$\Game$	2061	\Game	$\diagdown$	231F	\diagdown
$\Bbbk$	207C	\Bbbk			

算术运算符号

$\dotplus$	1275	\dotplus	$\ltimes$	226E	\ltimes
$\smallsetminus$	2272	\smallsetminus	$\rtimes$	226F	\rtimes
$\doublecap$	1265	\doublecap (U)	$\leftthreetimes$	1268	\leftthreetimes
$\doublecup$	1264	\doublecup (U)	$\rightthreetimes$	1269	\rightthreetimes
$\barwedge$	125A	\barwedge	$\curlywedge$	1266	\curlywedge
$\veebar$	1259	\veebar	$\curlyvee$	1267	\curlyvee
$\doublebarwedge$	125B	\doublebarwedge			
$\boxminus$	120C	\boxminus	$\circleddash$	127F	\circleddash
$\boxtimes$	1202	\boxtimes	$\circledast$	127E	\circledast

⊡ 1200 \boxdot
⊞ 1201 \boxplus
⋇ 223E \divideontimes
◎ 127D \circledcirc
· 1205 \centerdot
⊺ 127C \intercal

逻辑关系符号

≦ 1335 \leqq
⩽ 1336 \leqslant
⪕ 1330 \eqslantless
≲ 132E \lesssim
⪅ 132F \lessapprox
≊ 2375 \approxeq
⋖ 236C \lessdot
⋘ 136E \lll,\llless
≶ 1337 \lessgtr
⋚ 1351 \lesseqgtr
⪋ 1353 \lesseqqgtr
≑ 132B \doteqdot,\Doteq
≓ 133A \risingdotseq
≒ 133B \fallingdotseq
∽ 1376 \backsim
⋍ 1377 \backsimeq
⫅ 136A \subseteqq
⋐ 1362 \Subset
⊏ 1340 \sqsubset
≼ 1334 \preccurlyeq
⋞ 1332 \curlyeqprec
≾ 132D \precsim
⪷ 2377 \precapprox
◁ 1343 \vartriangleleft
⊴ 1345 \trianglelefteq
⊨ 130F \vDash
⊪ 130E \Vvdash
⌣ 1360 \smallsmile
⌢ 1361 \smallfrown
≏ 136C \bumpeq
≎ 136D \Bumpeq
∝ 135F \varpropto
◀ 134A \blacktriangleleft

≧ 133D \geqq
⩾ 133E \geqslant
⪖ 1331 \eqslantgtr
≳ 1326 \gtrsim
⪆ 1327 \gtrapprox
⋗ 236D \gtrdot
⋙ 136F \ggg,\gggtr
≷ 133F \gtrless
⋛ 1352 \gtreqless
⪌ 1354 \gtreqqless
≗ 1350 \eqcirc
≗ 1324 \circeq
≜ 132C \triangleq
∼ 2373 \thicksim
≈ 2374 \thickapprox
⫆ 136B \supseteqq
⋑ 1363 \Supset
⊐ 1341 \sqsupset
≽ 133C \succcurlyeq
⋟ 1333 \curlyeqsucc
≿ 1325 \succsim
⪸ 2376 \succapprox
▷ 1342 \vartriangleright
⊵ 1344 \trianglerighteq
⊩ 130D \Vdash
∣ 2370 \shortmid
∥ 2371 \shortparallel
≬ 1347 \between
⋔ 1374 \pitchfork
϶ 237F \backepsilon
▶ 1349 \blacktriangleright

∴ 1329 \therefore ∵ 132A \because

否定关系符号

≮ 2304 \nless ≯ 2305 \ngtr
≰ 2302 \nleq ≱ 2303 \ngeq
⩽̸ 230A \nleqslant ⩾̸ 230B \ngeqslant
≦̸ 2314 \nleqq ≧̸ 2315 \ngeqq
⪇ 230C \lneq ⪈ 230D \gneq
≨ 2308 \lneqq ≩ 2309 \gneqq
≨ 2300 \lvertneqq ≩ 2301 \gvertneqq
⋦ 2312 \lnsim ⋧ 2313 \gnsim
⪉ 231A \lnapprox ⪊ 231B \gnapprox
⊀ 2306 \nprec ⊁ 2307 \nsucc
⋠ 230E \npreceq ⋡ 230F \nsucceq
⪵ 2316 \precneqq ⪶ 2317 \succneqq
⋨ 2310 \precnsim ⋩ 2311 \succnsim
⪹ 2318 \precnapprox ⪺ 2319 \succnapprox
≁ 231C \nsim ≇ 231D \ncong
∤ 232E \nshortmid ∦ 232F \nshortparallel
∣ 232D \nmid ∦ 232C \nparallel
⊬ 2330 \nvdash ⊭ 2332 \nvDash
⊮ 2331 \nVdash ⊯ 2333 \nVDash
⋪ 2336 \ntriangleleft ⋫ 2337 \ntriangleright
⋬ 2335 \ntrianglelefteq ⋭ 2334 \ntrianglerighteq
⊈ 232A \nsubseteq ⊉ 232B \nsupseteq
⫅̸ 2322 \nsubseteqq ⫆̸ 2323 \nsupseteqq
⊊ 2328 \subsetneq ⊋ 2329 \supsetneq
⫋ 2320 \varsubsetneq ⫌ 2321 \varsupsetneq
⫋ 2324 \subsetneqq ⫌ 2325 \supsetneqq
⫋ 2326 \varsubsetneqq ⫌ 2327 \varsupsetneqq

箭头

⇇ 1312 \leftleftarrows ⇉ 1313 \rightrightarrows
⇆ 131C \leftrightarrows ⇄ 131D \rightleftarrows
⇚ 1357 \Lleftarrow ⇛ 1356 \Rrightarrow
↞ 1311 \twoheadleftarrow ↠ 1310 \twoheadrightarrow
↢ 131B \leftarrowtail ↣ 131A \rightarrowtail
↫ 1322 \looparrowleft ↬ 1323 \looparrowright

⇋ 130B \leftrightharpoons
⇌ 130A \rightleftharpoons
↶ 2378 \curvearrowleft
↷ 2379 \curvearrowright
↺ 1309 \circlearrowleft
↻ 1308 \circlearrowright
↰ 131E \Lsh
↱ 131F \Rsh
⇈ 1314 \upuparrows
⇊ 1315 \downdownarrows
↿ 1318 \upharpoonleft
↾ 1316 \upharpoonright,\restriction
⇃ 1319 \downharpoonleft
⇂ 1317 \downharpoonright
⊸ 1328 \multimap
⇝ 1320 \rightsquigarrow
↭ 1321 \leftrightsquigarrow

否定箭头

↚ 2338 \nleftarrow
↛ 2339 \nrightarrow
⇍ 233A \nLeftarrow
⇏ 233B \nRightarrow
↮ 233D \nleftrightarrow
⇎ 233C \nLeftrightarrow

第四章　LaTeX 简介

第一章和第二章介绍了 TeX 的基本使用方法，其中包括 plain 格式、vanilla 格式和 ltr 格式。第三章介绍了美国数学学会用 TeX 宏命令编写的 $\mathcal{AMS}$-TeX 程序包，简化了数学文章的源文件编写。虽然 $\mathcal{AMS}$-TeX 定义了一些容易理解的 TeX 命令和一些数学符号与字体，方便了普通人编写源文件，但是 $\mathcal{AMS}$-TeX 的重点还只是在于如何排版的层次上，没把 TeX 的一些重要功能定义成 TeX 命令，因此要实现诸如目录，交叉索引，公式自动编号等功能需要高级 TeX 排版和程序设计人员。

正是由于这种原因，美国计算机学家 Leslie Lamport 在 20 世纪 80 年代初期开发了 LaTeX（该标记在源文件中用 \LaTeX 表示） 程序包，提供了一组生成复杂文档所需的高级命令。利用该程序包，即使没有排版和程序设计知识的使用者也可以充分发挥 TeX 所提供的强大功能。LaTeX 可以在作者根本不知道所以然的条件下，自动给出章节、参考文献、浮动图表、数学公式等的编号，还提供了在全文的任何地方引用这些编号及其所对应对象所在页码的方法。

LaTeX 和其它软件一样，也定期地进行更新和修订。现在的版本号已固定为 2.09 ，而修订只是用日期来区分。最近的一次大修订发生在 1991 年 12 月 1 日，其后直到 1992 年 3 月 25 日为止，还有几次小的修订。为了尝试建立一个真正的改进标准，在 1989 年，由 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命令集合，这些命令是来自于各种软件包为了实现某一目的而设计出来的。

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。然而，这一计划尚未完成。作为阶段性成果，在 1994 年发行了 LaTeX 2_ε 及出版了 Lamport 基本手册第二版。本章简单介绍 LaTeX 2_ε 源文件的编写及编译方法。

§4.1　LaTeX 2_ε 源文件的组织构成与编译

与 $\mathcal{AMS}$-TeX源文件类似， LaTeX 2_ε 也是在 plain 格式的基础上添了一些便于排版的控制序列而构成，因此它具备了 plain 格式的基本特性。在本节中介绍 LaTeX 2_ε 源文件的组织构成和编译。

§4.1.1　LaTeX 2_ε 源文件的组织构成

采用 LaTeX 2_ε 的源文件主要由两大部分构成。第一部分是文章版面定义部分，第二部分是正文部分。源文件的具体结构如下：

```
\documentclass[ 选项 1 , 选项 2 ,  ...]{ 类 }
\usepackage[ 选项 1 , 选项 2 ,  ...]{ 宏包 1 , 宏包 2 ,  ...}
```

```
...
...   } 文章的版面定义部分
...
\begin{ducument}
...
...   } 文章的正文部分
...
\end{document}
```

在 LaTeX 2ε 源文件中，\begin{document} 标志着正文的开始，\end{document} 标志着源文件的结束（ LaTeX 2ε 总是忽略 \end{document} 之后的任何字符）。我们将在下面几节中介绍源文件各个部分的编写。

§4.1.2 LaTeX 2ε 源文件的编译

在 TeX 软件包中都含有编译 LaTeX 源文件的批处理文件 latex.bat（或程序 Latex.exe ）。为把 LaTeX 源文件编译成 .dvi 文件，只需键入

```
latex [d:][path]filename↩
```

其中 [d:] 表示源文件所在的驱动器，[path] 表示源文件所在的路径，filename 表示源文件的全名。当源文件的扩展名为 .tex 时可以省略其扩展名；当源文件就位于当前目录下时可以省略方括号部分的输入。

运行 latex 的结果会产生一些诸如 .dvi 、 .log 、 .aux 等文件。有了 .dvi 文件之后就可以按照第 1 页 §1.1 节中介绍的方法显示或打印输出排版结果。另外要注意的是当源文件使用了 LaTeX 2ε 的交叉引用或自动建立目录等功能时，对源文件连续编译两遍才能正确地显示和打印编译出的 .dvi 文件。

§4.2 LaTeX 2ε 源文件中的版面定义部分

本节将简单介绍类的选项及其它一些控制版面的方法。

§4.2.1 LaTeX 2ε 源文件中的类和类选项

在第 105 页 §4.1.1 节中介绍的 LaTeX 2ε 源文件结构中，

```
\documentclass[ 选项 1 , 选项 2 ,  ...]{ 类 }
```

是决定源文件所采用类及选项的结构．其中类的可能取值有如下几种 ：

book、report、article 或 letter

这些类之间的差异在于排版格式的不同： book 类用于书的排版，它可以有目录、章、节等部分，能够根据页面的内容安排页眉的内容； report 类用于学术报告的排版，它也可以有章、节等部分； article 类用于论文的排版，它可以有 parts（部分）， abstract（摘要）， section（节）， subsection（小节）等； letter 类用于信件的排版。一般来讲， 10 页至 20 页的短文用 article 类，分章节的长篇报告用 report 类，写书用 book 类，写信用 letter 类。类可以有多种选项，包括字体的尺寸、纸张的大小、页面的栏数等，下面分别加以介绍。

1. 可供选择的基本字体尺寸如下： 10pt ， 11pt ， 12pt 。这里 10pt 是 LaTeX 2ε 的默认字体尺寸， 11pt 的尺寸是 10pt 尺寸的 1.1 倍， 12pt 的尺寸是 10pt 尺寸的 1.2 倍。注意，这里尺寸的改变可以使文章中所有的字体（包括标题，节标题，正文字体等）按比例放大或缩小。例如

```
\documentclass[12pt]{book}
```

表示源文件采用书类排版，其正文的字体为 12pt 。

2. 可供选择的纸张类型如下 6 种：

letterpaper	11 × 8.5in
legalpaper	14 × 8.5in
executivepaper	10.5 × 7.25in
a4paper	29.7 × 21cm
a5paper	21 × 14.8cm
b5paper	25 × 17.6cm

其中左边列是可供选择的参数，右边列是对应参数所定义的纸张的实际尺寸。LaTeX 2ε 默认的纸张是美国信纸，即 letterpaper 选项。例如

```
\documentclass[11pt,b5paper]{article}
```

表示采用论文类排版，其中正文部分的字体为 11pt ，所用的纸张为 B5 复印纸。

3. 纸张打印的方向： LaTeX 2ε 默认的是纵向模式，即纸张中长的方向是打印机进纸的方向。使用选项

```
landscape
```

使纸张短的方向成为打印机进纸的方向（即横向模式）。例如

```
\documentclass[landscape]{article}
```

表示使用横向模式排版论文。

4. 页面排版栏数：类选项

```
onecolumn
```

表示使用单栏排版，这是 LaTeX 2ε 默认的选项。如果想选用双栏排版，使用类选项：

```
twocolumn
```

当采用此选项时，还可以使用控制词 \columnsep{...} 和 \columnseprule 来指定两栏之间的距离和竖线的宽度。在默认情况下，竖线的宽度为 0 ，即不打印两栏之间的竖线。例如

```
\documentclass[twocolumn]{article}
```

表示用双栏排版论文。又如

```
\documentclass[twocolumn]{article}
\setlength{\columnsep}{0.5cm}
```

表示用双栏排版论文，并且两栏之间的距离为 0.5cm 。再如

```
\documentclass[twocolumn]{article}
\setlength{\columnseprule}{0.1cm}
```

表示用双栏排版论文，并且两栏之间有一条宽度为 0.1cm 的竖线。

5. 页码打印方式：如果想使页码在所有的页中打印在相同的位置，使用选项：

```
oneside
```

这是 article 类和 report 类的默认选项。如果想使页码在奇数页打印在右边，偶数页打印在左边，使用选项：

```
twoside
```

这是 book 类的默认选项。

6. book 类中各章起始页的选项：在 book 类中，默认选项是每一章都从奇数页开始，也就是说各章都是从右边页开始。这样在前面一章的结束页码为奇数时，LaTeX 2ε 会在新的一章开始前输入一个空白页，以保证新的一章是从奇数页开始。为了改变这一特性，使得新的一章总是在前一章的下一页开始打印，可使用选项

```
openany
```

如果想使新的一章总是从右边页开始，可以用类选项

```
openright
```

7. book 类和 report 类中的标题选项：这两个类默认的选项是把标题作为单独的一页来打印。如果想要标题和接下来的文字排印在同一页上，可以使用选项：

```
notitlepage
```

8. article 类中的标题选项：该类的默认选项是把标题和接下来的文字排印在同一页上。如果想把标题作为单独的一页来打印，可以使用选项：

```
titlepage
```

9. 单列公式的编号选项：LaTeX 2ε 默认的是把单列公式的编号放在公式的右边。如果想把单列公式的编号放在公式的左边，可以使用选项：

```
leqno
```

10. 单列公式对齐方式的选项：LaTeX 2ε 默认的是把单列公式按中心对齐。如果想把单列公式按左边对齐，可以使用选项：

```
fleqn
```

接下来可以用控制词 \mathindent 来确定单列公式的左边缩进量的大小。例如

```
\documentclass[fleqn]{article}
\setlength{\mathindent}{2cm}
```

表示选择 article 类打印论文，并且使所有的单列公式都左边对齐，对齐线位于正文左边界缩进 2cm 。

11. 标识排版超出正文右边界的选项：如果 LaTeX 2ε 在编译源文件时发现一行超出了规定版面的右边界，它会给出警告信息并继续编译，你可以在 .log 文件中找到相应的信息。为了确定超出边界的行，LaTeX 2ε 可以在该行的右边打印一个粗黑条来标识，为此需要选项

    draft

    如果不需要用黑条标识超宽的行，可以应用选项

    final

    这是 LaTeX 2ε 的默认选项。

12. 参考文献的排列方式选项：在默认的情况下，如果一项参考文献有两行以上的内容，LaTeX 2ε 会把这一项参考文献各行的左边界对齐；也可以让 LaTeX 2ε 把该项参考文献从第二行开始后的各行的左边界缩进一个量来对齐，为此需要选择类选项

    openbib

注意，前面提到的像 \mathindent、\columnsep 和 \columnseprule 都是表示长度单位的量，可以在源文件的任何地方使用 112 页 §4.2.4 小节提供的命令修改它们所代表的实际长度。修改上面提到的这些长度单位所代表的实际长度的命令可以在源文件中多次使用，其作用范围是直到下一次修改同一长度单位所代表的长度。

§4.2.2 LaTeX 2ε 源文件中宏程序包的调用

虽然 LaTeX 2ε 类规定了文件的版面安排方式，但有时人们会对这种规定的方式做一些修改以满足特殊需要。为了方便使用者，一些团体和个人编写了可重复利用的宏程序包，用来修改 LaTeX 2ε 类所规定的排版方式，甚至定义一些新的命令。为了简便，下面把宏程序包简称为宏包。LaTeX 2ε 宏包的扩展名都是 .sty，这和前三章介绍的格式文件的扩展名相同，实际上宏包就是一种专用于 LaTeX 2ε 的格式文件。

通常是在源文件的版面定义部分调用宏包，具体调用的命令格式为：

```
\usepackage[ 选项 1 , 选项 2 ,  ...]{ 宏包 1 , 宏包 2 ,  ...}
```

例如

```
\usepackage{Amsfonts,amssymb}
```

表示调用宏包 Amsfonts.sty 和 amssymb.sty，这两个宏包定义了美国数学会所建立的字体和符号的控制词。

注意，调用宏文件命令 \usepackage 中的选项适用于在该命令中选取的所有宏包，如果其中一个宏包不能理解选项命令，在编译这个源文件时会给出一条警告信息。另外，如果仅想使用类默认的版面定义，可以在源文件中不使用控制词 \usepackage。

§4.2.3 修改 LaTeX 2ε 类规定的版面格式

LaTeX 2ε 允许对类所规定的版面格式做出修改，这里的版面格式包括页眉、页脚、正文的长和宽、行距、自然段第一行的缩进量等。

修改页面格式的控制词为：

```
\pagestyle{ 格式名 }
```

和

```
\thispagestyle{ 格式名 }
```

其中“格式名”可以是 plain、empty、headings 或 myheadings，我们将要在下面几个小段中分别介绍这些格式。控制词 \pagestyle 和 \thispagestyle 的用法类似，只是后者的作用范围是该命令出现的页。

§4.2.3.1 LaTeX 2ε 中的 plain 格式

plain 格式的页眉是空的，页脚由居中的页码组成。LaTeX 的默认格式是 plain 格式。控制词

```
\pagestyle{plain}
```

表示从此之后采用 plain 格式。当 LaTeX 读到这个命令后，就开始使用 plain 格式排版。控制词

```
\thispagestyle{plain}
```

表示当前页采用 plain 格式。

§4.2.3.2 LaTeX 2ε 中的 empty 格式

empty 格式的页眉和页脚是空的。控制词

```
\pagestyle{empty}
```

表示从此之后采用 empty 格式。控制词

```
\thispagestyle{empty}
```

表示当前页采用 empty 格式。

§4.2.3.3 LaTeX 2ε 中的 headings 格式

headings 格式的页眉是根据源文件所采用类的不同而有所变化。对于具有 twoside 选项的 article 类，它的左边纸的页眉为节的标题，右边纸的页眉为小节的标题；对于具有 twoside 选项的 book 类或 report 类，它的左边纸的页眉为章的标题，右边纸的页眉为节的标题；对于具有 oneside 选项的 article 类，它的眉为节的标题；对于具有 oneside 选项的 book 或 report 类，它的页眉为章的标题。命令

```
\pagestyle{headings}
```

表示从此之后采用 headings 格式。

§4.2.3.4 LaTeX 2ε 中的 myheadings 格式

myheadings 格式是需要自己定义页眉文字的格式，其页眉文字作为控制词 \markright 或 \markboth 的参量。\markright 可以修改所有右边页的页眉；而 \markboth 用于有 twoside 选项类的源文件中同时修改左边页和右边页的页眉。由于在类的 oneside 选项中认为所有的页都为右边纸，所以此时可以用 \markright 修改所有页的页眉。这两个控制词的用法如下：

```
\markright{ 右边纸页眉 }
\markboth{ 左边纸页眉 }{ 右边纸页眉 }
```

例如

```
\documentclass{article}
\pagestyle{myheadings}
.............
\markright{My Name}
```

表示从 \markright{My Name} 之后把右边纸的页眉打印成 My Name。又如

```
\documentclass[twoside]{article}
\pagestyle{myheadings}
.............
\markboth{Left Page}{My Name}
```

表示从 \markboth{Left Page}{My Name} 之后把左边纸的页眉打印成 Left Page，把右边纸的页眉打印成 My Name。

§4.2.3.5 LaTeX 2ε 中的页码编号

LaTeX 2ε 的页码编号可以取阿拉伯数字，小写罗马数字，大写罗马数字，小写字母以及大写字母。选何种编号由下面的命令所决定：

```
\pagenumbering{ 编码方式 }
```

其中“编码方式”可以取： arabic（阿拉伯数字）、 roma（小写罗马数字）、 Roma（大写罗马数字）、 alpha（小写字母）以及 Alpha（大写字母）。另外，使用这个命令时， LaTeX 2ε 自动把当前的页码重置为 1。例如

```
\pagenumbering{roma}
```

会把当前页的页码打印成为小写罗马数字 i，其后面各页的页码将依次用小写罗马数字递增打印。

还可以用命令设置当前页码的数值，其方法如下：

```
\setcounter{page}{ 页码 }
```

其中“页码”代表当前页的页码数字。例如

```
\setcounter{page}{5}
```

表示把当前的页码数字设置成 5.

§4.2.3.6 LaTeX 2ε 中与段落有关或与页面有关的控制

在 LaTeX 2ε 中定义了一些表示特定版面含意的长度单位或量，它们所代表的长度或大小可以通过一些特定的命令来赋值，详见 §4.2.4。下面列出的是常用的长度单位及相应的含义：

\parskip	两个自然段落之间的距离
\parindent	自然段第一行起始位置的缩进量
\baselineskip	行间距
\textwidth	正文的宽度，即版面的宽度
\textheight	正文的高度，即版面的高度
\oddsidemargin	奇数页正文左边界的位置
\evensidemargin	偶数页正文左边界的位置

\topmargin	从上纸边到页眉顶部的位置
\headheight	页眉的高度
\headsep	页眉基线到版面顶部的距离
\topskip	从版面顶部到正文第一行基线的距离
\footskip	正文版面底部与页脚底部的距离

如何改变这些长度单位所代表长度的例子见 §4.2.4。注意上面的表示长度单位的命令中，除了前面三个所代表的长度可以用在源文件中的任何地方修改外，其它的命令最好在文章的版面定义部分（即控制词 \begin{document} 之前的部分）赋值。还要注意，LaTeX 2ε 是从纸张上边的 1 英寸开始计算纵向长度，从纸张左边的 1 英寸开始计算横向长度。

还有两个有用的长度单位

\paperwidth	纸张的宽度
\pagerheight	纸张的高度

他们的长度都是在类的声明中作为类选项所定义的，一般我们不会在其后的源文件中修改它们。LaTeX 2ε 主要用他们来计算页边界，从而使页面的位置居中。

另外，还有一个控制正文行距的数值量

\baselinestretch

它表示相邻两行基线之间的距离因子。这个命令表示的是一个十进位小数而不是一个长度，它的初始值为 1，表示标准的基线距离。如果给它的赋值为 1.5（具体的赋值方法见 §4.2.4），则表示相邻两行的基线之间的距离为标准基线距离的 1.5 倍，即比标准基线距离多 0.5 倍的距离。

TeX 中定义了各种不同的表示长度的单位，在第 9 页第 §1.2.8 小节中介绍各种长度单位的含义同样适合于 $\mathcal{AMS}$-TeX 和 LaTeX 2ε 。

对于 book 类或 twoside 选项，每一页正文的底边恰处在相同的位置上，而对于其他的类或者 oneside 选项，则各页正文的底边会稍有变化。使用者可以随时调用命令

\flushbottom

使 LaTeX 2ε 开始把正文版面底边安排在相同的位置上；也可以随时调用命令

\raggedbottom

使 LaTeX 2ε 开始把正文版面的底边位置的安排稍有变化。

在 LaTeX 2ε 中，在一个自然段的前面可以使用命令

\indent

\noindent

分别表示该自然段的第一行的头一个字符是否缩进由 \parindent 所代表的长度。

§4.2.4 LaTeX 2ε 中特定长度单位或量的赋值方法

在 LaTeX 2ε 中由一些修改长度单位或量的命令，本小节中仅介绍其中一些常用命令。需要注意的是有些长度单位（像 \paperwidth 和 \paperheight 等）所代表的长度不能用本节中的命令修改；有些长度单位所代表的长度只能在版面定义部分修改；还有些长度单位（像 \parskip、\parindent 和 \baselineskip 等）所代表的长度能在源文件的地方任何修改。对于可以在源

文件的任何地方修改其所代表的实际长度的长度单位，其作用范围是直到下一次修改该长度单位所代表的长度。

§4.2.4.1 LaTeX 2ε 中特定长度单位的赋值

可以用

```
\setlength{ 长度单位 }{ 长度 }
```

来设置特定长度单位所代表的长度到给定的长度，其中“长度单位”是 LaTeX 2ε 中定义的表示长度单位的控制词，“长度”是具体的长度值及单位或者是具体的长度值及表示长度单位的控制词。

例如，

```
\setlength{\parskip}{2cm}
```

表示在该命令后把自然段间的距离定义为 2cm；而

```
\setlength{\oddsidemargin}{0.5\evensidemargin}
```

表示在该命令后把奇数页正文的左边界设置成 \evensidemargin 所代表的长度的 0.5 倍。

另外，LaTeX 2ε 是从纸张上边的 1 英寸开始计算纵向长度，从纸张左边的 1 英寸开始计算横向长度。因此

```
\setlength{\oddsidemargin}{0in}
```

表示正文的左面有 1 英寸的空白，而

```
\setlength{\topmargin}{0in}
```

表示页眉的上方有 1 英寸的空白。

§4.2.4.2 LaTeX 2ε 中特定长度单位的修改

为增加或减少特定长度单位所代表的长度，可以用

```
\addtolength{ 长度单位 }{ 长度 }
```

其中“长度单位”是 LaTeX 2ε 中定义的表示长度单位的控制词，“长度”是具体增加的长度。

例如，

```
\addtolength{\parskip}{0.1cm}
```

表示把自然段之间的距离增加 0.1cm；而

```
\addtolength{\parskip}{-0.1cm}
```

表示把自然段之间的距离减少 0.1cm。

§4.2.4.3 LaTeX 2ε 中设置特定长度单位到字符串的长度

为了便于长度单位的设置，LaTeX 2ε 还提供了把长度单位所代表的长度设置成一行字符串的自然长度的命令，具体实现方式如下：

```
\settowidth{ 长度单位 }{ 字符串 }
```

其中“长度单位”是 LaTeX 2ε 中定义的表示长度单位的控制词，“字符串”是一段文本。

例如，

```
\settowidth{\indent}{we}
```

表示把自然段第一行的缩进量设置成英文单词 we 的宽度。

§4.2.4.4 LaTeX 2ε 中设置基线间距离的方法

在 LaTeX 2ε 中，可以用如下的命令结构设置正文文本中相邻两行基线间的距离：

```
\renewcommand{\baselinestretch}{ 数值 }
```

其中“数值”是一个小数。

例如，

```
\renewcommand{\baselinestretch}{1.5}
```

表示把相邻两条基线的距离增加到标准基线距离的 1.5 倍，即相邻两行文本间的基线距离比标准基线距离多 0.5 倍。

§4.3 LaTeX 2ε 源文件中的正文部分

LaTeX 2ε 源文件中的正文部分是由命令 \begin{document} 开始，由命令 \end{document} 结束。在正文部分，TeX 所定义的绝大多数控制序列都可以使用。在 LaTeX 2ε 中定义了众多的环境结构，以方便排版。在本节中分几小节介绍正文的建立。

§4.3.1 LaTeX 2ε 源文件中的环境

一个环境的作用是对其内部的文本做特殊（即不同于正文）的处理，以达到特定的排版效果。环境的说明具有如下的格式：

```
\begin{ 环境变量 }
  .........
\end{ 环境变量 }
```

其中“环境变量”代表 LaTeX 2ε 所定义的一个环境变量名称，“ ... ”代表环境要处理的文字。通常环境变量的英文名称和环境变量的含义相关联；例如

```
\begin{document}
  .........
\end{document}
```

表示正文环境，这里的英文单词 document 恰是文件、文档的意思；

```
\begin{math}
  .........
\end{math}
```

表示正文中的数学公式环境，这里的 math 正是英文单词 mathematics 的缩写，表示数学的意思。

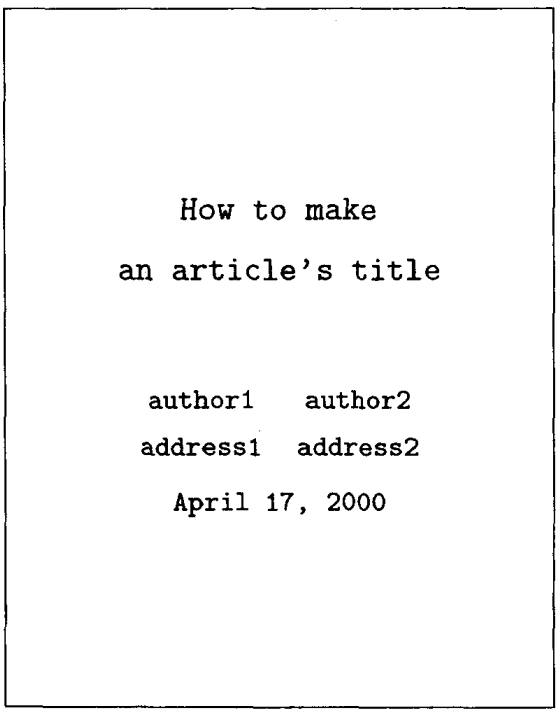
§4.3.2 LaTeX 2ε 中的文章标题、作者和写作日期

在 LaTeX 2ε 中可以用带参量的控制词方法输入文章的标题、作者和文章的完成日期。在源文件中，文章的标题和作者分别用如下的结构：

```
\title{...\\...\\...}
\author{***\\***\\***}
\maketitle
```

其中 ... 为文章的标题的文本内容，*** 为作者及地址的文本内容，\\ 表示另外起一行。另外，多作者的输入可以在两个作者的姓名地址之间使用控制词 \and ，使它们并排放在文章的标题的下面。下面给出一个具体的例子。

```
\title{ How to make \\
an article's title}
\author{author1 \\ address1
\and author2 \\ address2}
\maketitle
```



这里要注意控制词 \maketitle 的作用，这个命令使得 LaTeX 2ε 完成标题、作者和文章完成的日期的排版。在上面的这个输入结构中，计算机会把当前机器系统的日期自动添加在作者地址的下面一行的中心位置。如果使用者想自己给出文章的写作日期，可以在控制词 \maketitle 的前面加上带参量的控制词

```
\date{...}
```

其中 ... 时表示日期的文本，此时标题、作者、地址及日期的输入结构变成下面的格式：

```
\title{...\\...\\...}
\author{***\\***\\***}
\date{...}
\maketitle
```

另外，还可以用 TeX 中控制词 \today 表示计算机系统时间。

如果想要两个作者及地址上下放置，可以把控制词 \and 换成换行符 \\ ，既可以按照如下的方式输入源文件：

```
\title{ How to make \\
an article's title}
\author{author1 \\ address1
```

```
\\ author2 \\ address2}
\maketitle
```

另外，还可以在文章的标题文本、作者文本或日期文本的任何地方使用带参量的控制词

```
\thanks{...}
```

来表示感谢、资助等，这里 ... 代表一段文本，LaTeX 2ε 将把这段文本作为脚注打印。

§4.3.3 LaTeX 2ε 中的摘要

可以用环境生成文章中的摘要，其具体格式如下：

```
\begin{abstract}
   ...
\end{abstract}
```

其中 ... 是输入摘要内容的地方。在 report 类中，摘要单独占一页，而且有页码；在 article 类中，默认的方式是摘要和文章的标题在同一页上；在 book 类中没有摘要。如果想在 article 类中使标题单独占一页，可以在类声明中使用类选项 titlepage。

§4.3.4 LaTeX 2ε 中的部分、章、节、段标题与自动编号

在 LaTeX 2ε 中，部分、章、节、段的标题可以实现自动编号，它们都可以用带参量的控制词来实现。本小节介绍这些标题在源文件中的表达方法以及自动编号的原理。

§4.3.4.1 LaTeX 2ε 中的部分、章、节、段标题表达方法

下面列出 LaTeX 2ε 中用于输入部分、章、节、段标题的控制词及含义：

\part{...}	部分标题
\chapter{...}	章标题
\section{...}	节标题
\subsection{...}	小节标题
\subsubsection{...}	子节标题
\paragraph{...}	段标题
\subparagraph{...}	小段标题

在上面的这些控制词中，除了 \part 不能自动编号外，其它的标题都可以有自动编号。这些控制词都是把参量作为相应的标题，例如本节的标题在源文件中可以表示如下：

```
\section{\LaTeXe\  源文件中的正文部分 }
```

本节标题前面的“ §4.3 ”是 LaTeX 2ε （该符号在源文件中用控制词 \LaTeXe 表示）根据其前面出现的章节数自动加上的。为了方便使用者在本文的其它地方引用本节的编号，我们还可以为本节的编号用带参量的控制词 \label{...} 添加一个标签。例如

```
\section{\LaTeXe\  源文件中的正文部分 \label{4-3}}
```

就表示本节编号的标签为“ 4-3 ”，这样就可以在书的其它任何地方用“ \ref{4-3} ”代表编号“ 4.3 ”。这种引用自动编号方法称作交叉引用，我们将在 §4.7 中详细介绍 LaTeX 2ε 中的交叉引用。

除了 \part 之外，表示章、节、段的控制词还有另外一种用法，就是同时给出相应的短标题，以供页眉和文件目录用。例如

```
\section[\LaTeXe\  的正文部分 ]{\LaTeXe\
 源文件中的正文部分 \label{4-3}}
```

表示 §4.3 节的标题“ LaTeX 2ε 源文件中的正文部分”的短标题为“ LaTeX 2ε 的正文部分”，这样由 LaTeX 2ε 自动生成的目录和页眉中就会使用这个短标题作为 §4.3 节的标题。下面给出在本书第一稿中一些章节标题在源文件中的表达方式：

```
\chapter{\LaTeX\  简介 \label{ch4}}
\section{\LaTeXe\  源文件的组织构成与编译 \label{4-1}}
\subsection{\LaTeXe\  源文件的编译 \label{4-1-2}}
\section{\LaTeXe\  源文件中的版面定义部分 \label{4-2}}
\subsection{ 类的选项 \label{4-2-1}}
\subsection{ 预定义格式文件的调用 \label{4-2-2}}
\subsection{ 修改  \LaTeXe\  类规定的版面格式 \label{4-2-3}}
\subsubsection{\LaTeXe\  中的 plain 格式 \label{4-2-3-1}}
\subsubsection{\LaTeXe\  中的 empty 格式 \label{4-2-3-2}}
\subsubsection{\LaTeXe\  中的页眉格式 \label{4-2-3-3}}
\subsubsection{\LaTeXe\  中的页码编号 \label{4-2-3-4}}
```

有时人们不想使用 LaTeX 2ε 为章、节、段标题自动编号，这时可以使用 * 格式的控制词，下面以控制词 \subsection 为例说明这种命令形式的表达方法。例如当我们不想把本小节的标题自动编号时，可以用如下的方式输入小节的标题：

```
\subsection*{\LaTeXe\  中的部分、章、节、段标题 }
```

为了表达方便，我们把上面形式的控制词称为 * 格式命令。注意，当在章、节、段的标题中使用这种 * 格式的控制词，相应的章、节、段的标题不会出现在由 LaTeX 2ε 自动生成的目录中。

除了 * 格式的控制词， LaTeX 2ε 可以把用前面介绍的控制词输入的章、节、段标题生成一个目录表，并把它记录在一个临时文件中。只需要几个简单的命令，就可以在源文件中表示这个目录，具体的实现方法在下一小节中介绍。

§4.3.4.2 LaTeX 2ε 中章、节、段标题的自动编号原理

下边简单介绍一下 LaTeX 2ε 实现章、节、段自动编号的原理。在 LaTeX 2ε 的内部，存在一系列的计数器，所有能够自动编号的命令（控制词）都对应着一个唯一的计数器。为了方便，我们称对应于 \chapter 的计数器为章计数器；称对应于 \section 的计数器为节计数器；称对应于 \subsection 的计数器为小节计数器；称对应于 \subsubsection 的计数器为子节计数器；称对应于 \paragraph 的计数器为段计数器；称对应于 \subparagraph 的计数器为小段计数器，LaTeX 2ε 把这些计数器中的值通过适当的组合就可以得到章、节、段中各种标题的自动编号。上面这些计数器是有级别的，它们的级别由大到小排列次序依次为：章计数器、节计数器、小节计数器、子节计数器、段计数器、小段计数器。当 LaTeX 2ε 看到前面提到的一个控制词时，就会把该控制词所对应的计数器中的数字加 1 ，同时把级别小于该计数器的所有计数器中的数字重置

为 0 ，还会往一些临时文件 .toc 中写入一些信息（ LaTeX 2ε 会利用这些信息生成目录表等结构）。 LaTeX 2ε 就是这样实现章、节、小节、子节、段、小段的自动编号。对于前面提到的 * 命令， LaTeX 2ε 不对任何计数器重新赋值，也不往临时文件中写入任何信息（除非有其它的 LaTeX 2ε 命令干预），由此读者可以想象到 * 命令对章、节、段自动编号以及自动产生的目录表的影响。 LaTeX 2ε 的自动编号及自动产生目录表的功能大大减轻了人们编写或修改源文件的劳动强度，减少了编号出错的可能性，这正是人们喜欢 LaTeX 2ε 的原因之一。

在默认的方式下， LaTeX 2ε 并不启动所有的章、节、段计数器。对于 book 类和 report 类， LaTeX 2ε 默认启动的计数器为章、节和小节计数器，这时它只能给章、节和小节自动编号。对于 article 类， LaTeX 2ε 默认启动的计数器为节、小节和子节计数器，这时它只能给节、小节和子节的标题自动编号。 LaTeX 2ε 启动的章、节、段计数器数是由深度所决定的。对于章、节、段计数器，章、节、段的深度定义如下： \section 的计数器的深度为 1 ，下一级别为 2 ，再下一级为 3 ， ... ， \subparagraph 的级别为 5 。 LaTeX 2ε 对于章、节、段计数器的默认深度为 2 ，这导致前提到的自动编号结果。可以用下面的结构设置深度：

```
\setcounter{scnumdepth}{...}
```

其中 ... 代表深度值，取值范围为 -1 到 5 之间的整数值。例如在源文件的版面定义部分输入

```
\setcounter{scnumdepth}{5}
```

可以使得所有的章、节、段计数器都打开，结果使所有章、节、段的标题都自动编号。

§4.3.5 LaTeX 2ε 中的目录

如果我们按照 §4.3.4 小节中介绍的命令编写各个章、节、段的标题，就可以让 LaTeX 2ε 自动生成一个目录表，该目录表包括各个章、节、段的编号和标题以及在文章中的页码，本书中的目录表就是这样生成的。 LaTeX 2ε 是通过命令

```
\tableofcontents
```

来插入目录表。 LaTeX 2ε 会在使用该命令的地方产生目录表。

如果不是特殊说明， LaTeX 2ε 不能把所有的章、节、段的标题收录到目录表中。在默认的情况下， book 类和 report 类只能把 \subsection 以上级别的标题收集到目录中；而 article 类则能把 \subsubsection 以上级别的标题收录到目录中。我们可以在源文件的版面定义部分用如下的命令改变 LaTeX 2ε 的自动建立目录的默认选项：

```
\setcounter{tocdepth}{...}
```

其中 ... 是章、节、段标题的深度。例如

```
\setcounter{tocdepth}{4}
```

表示要把 \paragraph 以上级别的章、节、段标题收录到目录表中。

还可以手工将一些条目添加到目录表中，其在源文件中的实现方法如下：

```
\addcontentsline{toc}{...}{\numberline {***}~~~}
```

其中 ... 可以取如下的单词之一： chapter 、 section 、 subsection 、 subsubsection 、 paragraph 、 subparagraph ， *** 可以是任何文本和字符，它作为标题文本前面的符号， ~~~ 是标题文本。例如，

```
\addcontentsline{toc}{subsection}{\numberline {5.1.5}title}
```

表示向目录中添加一子标题项，标题的题目为 title ，标题前的编号为 5.1.5 ，该标题对应的页码为本页的页码。

我们还可以通过下面的结构向目录中添加任何文本、控制词、控制符等：

```
\addtocontents{toc}{...}
```

其中 ... 表示要添加的内容。例如

```
\addtocontents{toc}{this is my test \protect\newpage}
```

表示向目录中添加如下文本 " this is my test \newpage "。

注意，当章、节、段的标题含有控制词时，可能会使自动生成的目录或页眉产生错误的排版。其原因是 LaTeX 2ε 不能向临时文件正确地传送控制词，解决的办法是在控制词前输入前缀命令 \protect ，使得其后面的控制词能够正确地传送到相应的临时文件中。

另外，要使显示或打印输出中的目录是最后的版本，必须连续编译两次源文件。这是因为在每次编译时 LaTeX 2ε 都会先读进临时文件 .toc 以建立本次编译的目录表，然后再更新目录表临时文件 .toc ，即在编译时只能用上次编译产生的目录表信息构造本次编译的目录表。

§4.3.6 LaTeX 2ε 中的附录

可以用 \appendix 命令来编写附录，具体的应用格式如下：

```
\appendix
  .........
```

其中 .... 代表附录文本。附录文本还可以包括章、节、段的标题，但是控制词 \appendix 使 LaTeX 2ε 重新设置了章或节等标题的自动编号表达方式及计数器的初始值，从而形成了附录所特有的风格。在 book 和 report 类中，章标题的自动编号变成了大写字母 A, B, C, ...，而节以下级别的各个标题的自动编号还是阿拉伯数字。在 article 类中，节标题的自动编号变成了大写字母 A, B, C, ...，而子节以下级别的各个标题的自动编号还是阿拉伯数字。附录还可以用如下的环境来实现：

```
\begin{appendix}
  .........
\end{appendix}
```

其中 .... 代表附录文本。

§4.3.7 LaTeX 2ε 中的参考文献

通常在科技图书、论文和报告中都会有参考文献，并且还要经常在正文中引用这些参考文献。参考文献通常都是由编号或是作者姓名的缩写等作为关键词来引用，如果我们发现了某个关键词有错误，那么修改起来将是十分麻烦的。虽然一般的编辑软件提供的查找替换功能可将关键词替换成为正确的结果，但是万一正文中的个别的关键词输入有误，比如说多个空格，或者少了个字母，这种办法就靠不住了。 LaTeX 2ε 提供了一种输入参考文献的方法，使得参考文献的引用和修改十分方便，本小节介绍这种方法。

在 LaTeX 2ε 中，参考文献是用如下的环境来生成：

```
\begin{thebibliography}{...}
   ***
\end{thebibliography}
```

其中 ... 为关键词样本， *** 表示各个不同的参考文献项。关键词样本表示的是最大关键词的宽度，它可以由一段文本组成，这段文本的宽度就定义为最大关键词的宽度。

每条参考文献项具有如下的结构：

```
\bibitem[...]{***}~~~
```

其中 ... 表示关键词，是正文中引用该条参考文献时（在方括号内）所显示的文本内容； *** 表示标签，所起作用是帮助在正文中表示关键词； ~~~ 表示该条参考文献的内容，通常包括作者、题目、出版社、年代、版本、页码等。在此处，关键词可以省略，即可以按如下的方式输入参考文献：

```
\bibitem{***}~~~
```

此时 LaTeX 2ε 用参考文献的自动编号作为关键词。

在正文中，参考文献通过如下的方式引用：

```
\cite{***}
```

其中 *** 是某项参考文献的标签， LaTeX 2ε 会再在该处显示相应参考文献的关键词文本，并把该文本置于方括号中。下面一段源文件是参考文献的例子：

```
\begin{thebibliography}{aaaaa67890123}
  \bibitem[Arnt]{a}Ant L., \textsl{An Introduction to Ant}, LiABC Co.,
  Inc., 1999
  \bibitem[Tiger]{t}Tag L., \textsl{An Introduction to Tiger}, LiABC Co.,
  Inc., 2000
  \bibitem[Mouse L., 2000]{n}Mouse L., \textsl{An Introduction to Mouse},
  LiABC Co., Inc.,  2000
\end{thebibliography}
```

在 book 类和 report 类中， LaTeX 2ε 把上面的这段参考文献另起一页打印成：

Bibliography

[Arnt] Ant L., *An Introduction to Ant*, LiABC Co., Inc., 1999

[Tiger] Tag L., *An Introduction to Tiger*, LiABC Co., Inc., 2000

[Mouse L., 2000] Mouse L., *An Introduction to Mouse*, LiABC Co., Inc., 2000

而在 article 类中， LaTeX 2ε 把上面的这段参考文献在同一页打印成：

References

[Arnt] Ant L., *An Introduction to Ant*, LiABC Co., Inc., 1999

[Tiger] Tag L., *An Introduction to Tiger*, LiABC Co., Inc., 2000

[Mouse L., 2000] Mouse L., *An Introduction to Mouse*, LiABC Co., Inc., 2000

在前面的参考文献例子中，LaTeX 2ε 会把 `\cite{n}` 打印成 `[Mouse L., 2000]`，把 `\cite{t}` 打印成 `[Tiger]`，把 `\cite{a}` 打印成 `[Arnt]`。

如果我们在参考文献中省略关键词项，即按如下的方式输入：

```
\begin{thebibliography}{a}
  \bibitem{a}Ant L., \textsl{An Introduction to Ant}, LiABC Co.,
  Inc., 1999
  \bibitem{t}Tag L., \textsl{An Introduction to Tiger}, LiABC Co.,
  Inc., 2000
  \bibitem{n}Mouse L., \textsl{An Introduction to Mouse},
  LiABC Co., Inc.,  2000
\end{thebibliography}
```

则对应的参考文献项的打印输出变成：

[1] Ant L., *An Introduction to Ant*, LiABC Co., Inc., 1999

[2] Tag L., *An Introduction to Tiger*, LiABC Co., Inc., 2000

[3] Mouse L., *An Introduction to Mouse*, LiABC Co., Inc., 2000

对应于这一段参考文献的输入，LaTeX 2ε 会把 `\cite{n}` 打印成 `[3]`，把 `\cite{t}` 打印成 `[2]`，把 `\cite{a}` 打印成 `[1]`。

§4.3.8 LaTeX 2ε 中编排书的源文件结构

在编写书的 book 类中，LaTeX 2ε 提供了几个特殊的命令，以方便该类源文件的编写，本小节介绍这些命令的用法。

我们在 §4.1.1 小节中已经介绍过 LaTeX 2ε 的源文件由版面定义部分和正文部分所构成，book 类又进一步把正文分成了几个小部分，其具体结构如下：

```
\begin{document}
\frontmatter
 ...
\mainmatter
 ...
\backmatter
 ...
```

```
\end{document}
```

在这个结构中，\frontmatter 和 \mainmatter 之间是输入书的前言和目录部分源文件的地方。一般书中这部分的页码是用罗马数字表示的，而控制词 \frontmatter 正是让 LaTeX 2ε 的页码用罗马数字表示。\mainmatter 和 \backmatter 之间是输入书的各章节内容的地方，控制词 \mainmatter 的作用是把页码重新设置成为阿拉伯数字 1，并且激活章、节、段的自动编号机制。\backmatter 和 \end{document} 之间是输入参考文献、索引、版本记录的地方，控制词 \backmatter 关闭了章、节、段的自动编号机制。

§4.4 LaTeX 2ε 中的数学公式

为了方便数学公式的排版，Donald Knuth 才开发出 TeX 系统。由于数学渗透于各个现代科研领域，才使得 TeX 如此流行。用 TeX 宏语言开发的 LaTeX 2ε 程序包，继承了 TeX 的强大数学排版功能，又定义了便于排版的环境和控制词，极大地方便了 TeX 源文件的编写。昔日需要 TeX 专家才能完成的排版任务，如今普通的科技人员可以利用 LaTeX 2ε 来完成。LaTeX 2ε 数学公式的表示方法兼容于 TeX 的方法，但是它提供了其它的一些表达数学公式环境的方法，还提供单列公式的自动编号及引用的功能等。一般来说，在 LaTeX 2ε 的数学环境中可以利用 TeX 的数学符号表达方法，但是 LaTeX 2ε 却缺少一些 $\mathcal{AMS}$-TeX 中定义的数学字体和符号。本节中主要介绍 LaTeX 2ε 中的数学公式，强调它与 $\mathcal{AMS}$-TeX 的区别，并阐明如何在 LaTeX 2ε 中使用 $\mathcal{AMS}$-TeX 中定义的字体和符号的方法。

§4.4.1 LaTeX 2ε 中的数学公式环境

在 LaTeX 2ε 中，第一种数学公式环境的表示方法是 TeX 的传统表示方法，即用美元符号 $ 来表示。在 LaTeX 的源文件中，正文中的数学公式可以用如下的结构表示：

```
$...$
```

其中 ... 表示组成数学公式的字符。例如正文公式 $x+y=z$ 在源文件中的输入为 `$x+y=z$` 。在 LaTeX 的源文件中，单列数学公式可以用如下的结构表示：

```
$$...$$
```

其中 ... 表示组成数学公式的字符。例如单列公式

$$x+y=z$$

在源文件中的输入为 `$$x+y=z$$` 。用美元符号 $ 表示数学公式的好处是比较可靠，但是这要求在源文件中美元符必须配对出现，万一出现该符号不能正确配对，则很难找到出错的地方。

在 LaTeX 2ε 中，第二种数学公式的表示方法是利用圆括号和方括号来表示。在 LaTeX 的源文件中，正文中的数学公式可以用如下的结构表示：

```
\(...\)
```

其中 ... 表示组成数学公式的字符.例如正文公式 $x+y=z$ 在源文件中的输入为 `\(x+y=z\)` 。在 LaTeX 的源文件中，单列数学公式可以用如下的结构表示：

```
\[...\]
```

其中 ... 表示组成数学公式的字符。例如单列公式

$$x+y=z$$

在源文件中的输入为 \[x+y=z\] 。利用圆括号和方括号表示数学环境的优点是更容易发现数学环境的设置错误。

在 LaTeX 2ε 中，第三种数学公式的表示方法是利用环境来表示。在 LaTeX 的源文件中，正文中的数学公式可以用如下的结构表示：

```
\begin{math}...\end{math}
```

其中 ... 表示组成数学公式的字符。例如正文公式 $x+y=z$ 在源文件中的输入为

```
\begin{math}
x+y=z
\end{math}
```

在 LaTeX 的源文件中，单列数学公式可以用如下的结构表示：

```
\begin{displaymath}...\end{displaymath}
```

其中 ... 表示组成数学公式的字符。例如单列公式

$$x+y=z$$

在源文件中的输入为

```
\begin{displaymath}
x+y=z
\end{displaymath}
```

用环境表示数学公式符合 LaTeX 2ε 的设计思想，环境的定义与英文单词的含义密切相关，使得人们便于记忆。

在默认的情况下， LaTeX 2ε 把单列数学公式排列在水平居中的位置。但是如果在类的选项中选择了 fleqn 选项， LaTeX 2ε 会把单列数学公式排列在左对齐的位置。在 fleqn 选项下，还可以用

```
\setlength{\mathindent}{...}
```

来选择单列数学公式的相对于正文左边界的左缩进量，其中 ... 为长度，表示缩进的量。例如在 fleqn 选项下

```
\setlength{\mathindent}{0.5cm}
```

表示文章的所有单列公式在距离正文左边界 0.5 厘米处对齐。

§4.4.2 LaTeX 2ε 中能够自动编号的单列数学公式环境

在 LaTeX 2ε 中，单列数学公式还可以用如下的结构表达：

```
\begin{equation}
\label{...}
***
\end{equation}
```

其中 ... 表示该数学公式的标签， *** 表示组成数学公式的字符。上面的这种单列数学公式环境的特殊之处是它还有一个自动的公式编号，而公式的编号可以用其标签表示。在默认的情况下， LaTeX 2ε 把自动公式编号放在单列公式的右边。但是当在类的选项中选取了 leqno 时，LaTeX 2ε 会把公式的自动编号放在单列公式的左边。例如

```
\begin{equation}
\label{f-1}
x*y=w
\end{equation}
```

对应的打印输出为：

$$x * y = w \tag{4.1}$$

而

```
\begin{equation}
\label{f-2}
x/y=u
\end{equation}
```

对应的输出为：

$$x/y = u \tag{4.2}$$

现在可以在源文件中用 (\ref{f-1}) 表示 (4.1)，用 (\ref{f-2}) 表示 (4.2)。

§4.4.3 LaTeX 2ε 中的多行单列数学公式环境

在 LaTeX 2ε 中，还有一个用于输入多行单列公式的环境：

```
\begin{eqnarray}
... & ... & ...\\
... & ... & ...\\
... & ... & ...
\end{eqnarray}
```

其中 ... 表示数学符号， & 表示公式的对齐位置， \\ 表示公式的换行。另外， LaTeX 2ε 会自动为上面多行单列数学公式的环境中每一行公式添加一个公式编号。例如

```
\begin{eqnarray}
x+y&=&z\label{my}\\
x/y& = &w\label{your}
\end{eqnarray}
```

对应的打印输出为：

$$x + y = z \tag{4.3}$$

$$x/y = w \tag{4.4}$$

现在我们也可以在源文件中分别用 (\ref{my}) 和 (\ref{your}) 表示公式的编号 (4.3) 和 (4.4)。

有时我们不想要多行单列公式的每一行都有一个编号，这可以通过控制词 \nonumber 来取消某一行的自动编号。例如

```
\begin{eqnarray}
 & &(x+y)^2+x^2-y^2 \nonumber \\
 &=&(x+y)^2+(x+y)(x-y) \nonumber \\
 &=&2x(x+y)
\end{eqnarray}
```

所对应的打印输出为：

$$\begin{aligned} & (x+y)^2+x^2-y^2 \\ = \ & (x+y)^2+(x+y)(x-y) \\ = \ & 2x(x+y) \end{aligned} \tag{4.5}$$

注意在表达上面公式的源文件中并没有设置标签，这并不影响该公式的自动编号功能，只是为你在其它地方引用该公式编号造成了困难。

如果我们不想对多行单列公式中的任何一行进行编号，则可以用如下的环境：

```
\begin{eqnarray*}
... & ... & ...\\
... & ... & ...\\
... & ... & ...
\end{eqnarray*}
```

这个环境的定义只是比多行单列公式环境的定义多了一个 * ，我们称它为 * 格式的多行单列公式环境。例如

```
\begin{eqnarray*}
 & &(x+y)^2+x^2-y^2 \\
 &=&(x+y)^2+(x+y)(x-y)  \\
 &=&2x(x+y)
\end{eqnarray*}
```

对应的打印输出为：

$$\begin{aligned} & (x+y)^2+x^2-y^2 \\ = \ & (x+y)^2+(x+y)(x-y) \\ = \ & 2x(x+y) \end{aligned}$$

§4.4.4 LaTeX 2ε 数学公式中的数学符号

虽然在 LaTeX 2ε 中大多数数学符号的表达方式和 TeX 中的完全相同，例如希腊字母、各种运算符号、各种逻辑关系符号等，但是在 LaTeX 2ε 中的一些数学结构的表达方法与 $\mathcal{AMS}$-TeX 或 TeX 的 vanilla 格式中的表达方法却有不同，例如矩阵、多行下标、分段函数等. 一般来说，在 TeX 中的表示数学符号的控制词可以直接在 LaTeX 2ε 中使用，而 TeX 中带有如下结构的表达方式不能在 LaTeX 2ε 中使用：

```
\... ***\end...
```

其中 ... 代表组成控制词的字母， *** 为该结构中的文字。在 LaTeX 2ε 源文件中，上面的这些结构都被环境所取代。例如， AMS-TeX 源文件中表示矩阵的结构 \matrix...\endmatrix 不能直接用在 LaTeX 2ε 源文件中。在本节中，主要介绍那些 LaTeX 2ε 中与 TeX 或 AMS-TeX 中表达方式不同的数学结构。

§4.4.4.1 LaTeX 2ε 数学公式中的 array 环境

AMS-TeX 源文件中表示矩阵元素排列的结构 \matrix...\endmatrix 不能用在 LaTeX 2ε 源文件中，这需要习惯于使用 AMS-TeX 的读者特别注意。在 LaTeX 2ε 源文件中矩阵元素的排列是利用如下的环境来实现：

```
\begin{array}[~]{...}
*** & *** \\
*** & *** \\
*** & ***
\end{array}
```

其中 [~] 为对齐方式参数，它表示矩阵与其外部同行其它数学符号或文本的对齐方式； ... 为列格式参数，它决定了矩阵的列数、对齐方式和列间空隙； *** 为数学符号； & 是矩阵新一列开始标志； \\ 是矩阵新一行开始的的标志。为了方便，我们称上面的环境为矩阵元素排列环境，简称为 array 环境。下面具体介绍 array 环境中对齐方式参数和列格式参数内容。

在 array 环境中，对齐方式参数只可以选择 t 或 b ，其中 t 代表与外部文本顶部对齐（这里的 t 实际是英文单词 top 的第一个字母）， b 代表与外部文本底部对齐（这里的 b 是英文单词 bottom 的第一个字母）。例如

```
$$x=\begin{array}[t]{cc}
a&b\\
c&d
\end{array}
=\begin{array}[b]{cc}
a&b\\
c&d
\end{array}
$$
```

对应的打印输出为

$$x = \begin{array}{cc} a & b \\ c & d \end{array} = \begin{array}[b]{cc} a & b \\ c & d \end{array}$$

其中第一个 array 环境的对齐方式参数 t 使其首行与该环境之外的等号在水平方向对齐，第二个 array 环境中的对齐方式参数 b 使其末行与该环境之外的数学符号在水平方向对齐。在一般的情况下，我们需要的是 array 环境与外部数学符号居中对齐，这可以通过省略对齐参数

来实现。例如源文件

```
$$x=\begin{array}{cc}
a&b\\
c&d
\end{array}
=\begin{array}{cc}
a&b\\
c&d
\end{array}
$$
```

对应的打印输出为

$$x=\begin{array}{cc} a & b \\ c & d \end{array} = \begin{array}{cc} a & b \\ c & d \end{array}$$

其中省略的两个对齐参数保证了源文件的两个 array 环境都是与其外部的数学符号以水平方向的中心对齐。

在 array 环境中的列格式参数不可以省略，这些参数中包括决定各个列对齐的参数，两列中间的分隔文本等。常用的表示列对齐的参数有 l 、 r 、 c 和 p{...} ，其中 l 代表列的左对齐（ l 为英文单词 left 的第一个字母）方式， r 代表列的右对齐（ r 为英文单词 right 的第一个字母）方式， c 代表列的中心对齐（ c 为英文单词 center 的第一个字母）方式， p{...} 代表文本方式的列，这里 ... 为文本的宽度。在 array 环境中， LaTeX 2ε 规定元素排列的列数等于对齐参数的个数。下面的公式

$$A=\begin{array}{cc} 111 & b \\ c & 1000 \end{array} = \begin{array}{ccc} 111 & b & 0.0001 \\ c & 1000 & 1 \end{array}$$

在源文件中的表达方式为：

```
$$A=\begin{array}{cc}
111&b\\
c&1000
\end{array}
=\begin{array}{lrc}
111&b&0.0001\\
c&1000&1
\end{array}
$$
```

其中第一个 array 环境中的两列以中心方式对齐，第二个 array 环境中的三个列分别为左对齐、右对齐和中心对齐。另外，第一个 array 环境中有两个对齐参数，所以它有两列；第二个

array 环境中有三个对齐参数，所以它有三列。又如

$$\begin{array}{p{3cm}cc}
\texttt{The first row} & b & 0.0001 \\
\texttt{The second row} & 1000 & 1
\end{array}$$

在源文件中的表达为

```
$$
\begin{array}{p{3cm}cc}
The first row&b&0.0001\\
The second row&1000&1
\end{array}
$$
```

其中的列格式化参数 p{3cm} 表示第一列为 3cm 宽的文本。

在 array 环境中， \\ 表示一行的结束，同一行中用 & 区分各个不同列的元素。同一行中， & 的个数必须等于 array 环境中元素的列数减 1 。另外，我们还可以在表示行结束的符号 \\ 的后面添加 [...] 控制该行与下一行之间的间距，这里 ... 为长度。例如

```
$$A=\begin{array}{cccc}
111&b&0.0001&10\\[1ex]
c&1000&1&1\\[-3ex]
2&3&4&5
\end{array}
$$
```

对应的输出为

$$A=\begin{array}{cccc}
111 & b & 0.0001 & 10 \\
c & 1000 & 1 & 1 \\
2 & 3 & 4 & 5
\end{array}$$

其中第一行和第二行之间的距离增加了 1ex ，第二行和第三行之间的距离减少了 3ex 。

§4.4.4.2 LaTeX 2_ε 数学公式中的 array 环境与矩阵和行列式

利用变尺寸圆括号、变尺寸绝对值符号和 array 环境容易表示矩阵和行列式。矩阵

$$\left(\begin{array}{cccc}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{array}\right)$$

在源文件中的表达为

```
$$
\left(
\begin{array}{cccc}
a_{11}&a_{12}&\cdots&a_{1n}\\
a_{21}&a_{22}&\cdots&a_{2n}\\
\vdots&\vdots&\ddots&\vdots \\
a_{m1}&a_{m2}&\cdots&a_{mn}
\end{array}
\right)
$$
```

行列式

$$\left|\begin{array}{cccc}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{array}\right|$$

在源文件中的表达为

```
$$
\left|
\begin{array}{cccc}
a_{11}&a_{12}&\cdots&a_{1n}\\
a_{21}&a_{22}&\cdots&a_{2n}\\
\vdots&\vdots&\ddots&\vdots \\
a_{m1}&a_{m2}&\cdots&a_{mn}
\end{array}
\right|
$$
```

§4.4.4.3 LaTeX 2ε 数学公式中的 array 环境与表格

在 §4.4.4.1 中，我们简单地介绍了 array 环境的对齐参数和列对齐方式参数。从 array 环境的输出结果来看，它很像一个由数组成的表，如果我们能够给它的各个行和列之间添加一些横线和竖线就可以构成一个表格。事实上，LaTeX 2ε 提供了构造这些横线和竖线的命令，本子节简单介绍这些命令。

在表格划线命令之前，先简单介绍一下与 array 环境用法基本相同的另外两个专门用来构造表格的环境。首先是 tabular 环境的构造：

```
\begin{tabular}[~]{...}
*** & *** \\
*** & *** \\
*** & ***
\end{tabular}
```

其中 ~ 、 ... 和 *** 的含义和用法完全与 §4.4.4.1 子节中 array 环境中的相同。另外一个构造表格的环境是 tabular* 环境，它只是比 tabular 环境多出了一个定义整个表宽的参数，其它的构造完全相同。 tabular* 环境的具体构造如下：

```
\begin{tabular*}{w}[~]{...}
*** & *** \\
*** & *** \\
*** & ***
\end{tabular*}
```

其中 w 是一个长度，它代表整个表的宽度，而 ~ 、 ... 和 *** 的含义和用法完全与 §4.4.4.1 子节中 array 环境中的相同。

在 LaTeX 2ε 中的 array、tabular 和 tabular* 环境的用法基本相同，在 §4.4.4.1 中介绍的参数设置方法对这三个环境都适用。这三个环境的的区别在于 array 环境仅能用在数学公式中，而 tabular 环境和 tabular* 环境则只能用在文本环境中；应用 tabular* 环境必须指定表格的总宽度，而 array 环境和 tabular 环境不必如此。下面以 array 环境为例介绍这三种环境通用的设置表格命令。

在 array 环境中，两列元素之间的间隔有如下的三种表示方法： |、|| 和 @{...}。其中 | 表示两列之间用一条竖直线作为分隔； || 表示两列之间用两条相邻竖直线作为分隔；@{...} 表示两列之间用一段文本来分隔，这里 ... 代表这段文本。例如

```
$$A=\begin{array}{l|r||c@{\quad 3\quad}c}
111&b&0.0001&9\\
c&1000&1&9
\end{array}
```

对应的打印输出为

$$A=\begin{array}{l|r||c@{\quad 3\quad}c}
111&b&0.0001&9\\
c&1000&1&9
\end{array}$$

注意，从打印输出来看应该有 5 列数字，其中第一列和第二列的后面分别有单竖线和双竖线作为列之间的分隔符，但实际上倒数第二列的文字在源文件中是作为列之间的分隔文本。分隔文本的灵活应用可以产生特殊的排版效果。另外，使用命令

```
@{\extracolsep{...}}
```

可以定义各列之间的空白，其中 ... 为列间隔距离，它的作用持续到下一次出现该命令。例如表达式

```
$$A=\begin{array}{@{\extracolsep{5ex}}lrc@{\extracolsep{1ex}}c}
111&b&0.0001&10\\
c&1000&1&1
\end{array}
$$
```

对应的打印输出为：

$$A=\begin{array}{cccc} 111 & b & 0.0001 & 10 \\ c & 1000 & 1 & 1 \end{array}$$

其中前三列的间隔空隙是 5ex ，最后两列之间的间隔空隙是 1ex 。间隔文本还可以用来取消两列之间的间隔空隙，为达到此目的，只需要在相应的两个列参数之间加入 @{} 即可。

还有一种简化表示相同的一组列参数的方法，其方式如下：

```
*{n}{...}
```

其中 ... 代表列格式化参数， n 代表把 ... 所代表的列格式化参数复制的份数。例如列格式化参数表达式 {*{5}{lr}} 等价于 {lrlrlrlrlr} 。例如

$$A=\left\|\begin{array}{c|c|c|c} 111 & b & 0.0001 & 10 \\ c & 1000 & 1 & 1 \end{array}\right\|$$

在源文件中的表达式为

```
$$A=\begin{array}{||@{}*{3}{c|}c@{}||}
111&b&0.0001&10\\
c&1000&1&1
\end{array}
$$
```

其中 array 环境中列格式参数中的第一个 || 代表输出中最左面的双竖线，其后的 @{} 表示取消第一列与左双竖线之间的空隙； *{3}{c|} 表示三个重复的 c| ，即 c|c|c| ；最后的 @{}|| 表示最右边的双竖线，并且最后一列元素与该双竖线之间没有空隙。

还可以用

```
\hline
```

为 array 环境添加横线。注意该命令只能用在第一个行开始的位置，并且连续两个这样的命令表示一条双横线。例如

$$A=\begin{array}{|c|c|c|c|} \hline 111 & b & 0.0001 & 10 \\ \hline c & 1000 & 1 & 1 \\ \hline 2 & 3 & 4 & 5 \\ \hline \end{array}$$

在源文件中的表达式为：

```
$$A=\begin{array}{||@{}*{3}{c|}c@{}||}
\hline
111&b&0.0001&10\\ \hline\hline
c&1000&1&1\\ \hline
2&3&4&5\\ \hline
\end{array}
$$
```

注意在这个 array 环境中的最后一条横线是通过在最后一行的后边添加 \\ \hline 得到，这里表示一个行结束的双反斜杠 \\ 不能省略。

还有一个在各个行之间添加横线的命令，它的主要特点是可以给指定的一行里的特定列下（上）面添加横线，该命令的具体格式为：

```
\cline{n-m}
```

其中 n 表示横线起始列， m 代表横线的结束列。例如

$$A=\begin{array}{||c|}{*{3}{c|}c@{}||}
\hline
111&b&0.0001&10\\ \hline\hline
c&1000&1&1\\ \cline{2-3}
2&3&4&5\\ \hline
\end{array}$$

在源文件中的表达为：

```
$$A=\begin{array}{||c|}{*{3}{c|}c@{}||}
\hline
111&b&0.0001&10\\ \hline\hline
c&1000&1&1\\ \cline{2-3}
2&3&4&5\\ \hline
\end{array}
$$
```

其中的 \cline{2-3} 表示在第三行中第二列和第三列的上面画横线。

类似于在某行的特定位置画横线，也可以在某列的特定位置画竖线。控制词

```
\vline
```

可以在某一行的元素之间画竖线。例如

$$A=\begin{array}{||c|}{*{3}{c}c@{}||}
\hline
111&b&0.0001&10\\ \hline\hline
c&\vline\quad 1000\quad&1&\vline 1\\ \hline
2&3&4&5\\ \hline
\end{array}$$

在源文件中的表达为

```
$$A=\begin{array}{||c|}{*{3}{c}c@{}||}
\hline
111&b&0.0001&10\\ \hline\hline
c&\vline\quad 1000\quad&1&\vline 1\\ \hline
2&3&4&5\\ \hline
\end{array}
$$
```

由此可以看到控制词 \vline 的打印效果。

§4.4.4.4 LaTeX 2ε 数学公式中的分段函数

数学公式中常用到分段函数的表达式，在 TeX 源文件中分段函数使用如下的结构来表达：

```
\cases...\\ ...\endcases
```

但是在 LaTeX 2ε 中上面的这个表达式就失效了。为了表示分段函数，可以借助于 array 环境。例如

$$f(x)=\begin{cases}-1, & \text{for } x<0\\ 0, & \text{for } x=0\\ 1, & \text{for } x>0\end{cases}$$

在 LaTeX 2ε 源文件中的表达为

```
\[
f(x)=\left\{
\begin{array}{ll}
-1,&\quad \mbox{for $x<0$}\\
0,&\quad \mbox{for $x=0$}\\
1,&\quad \mbox{for $x>0$}
\end{array}
\right.
\]
```

其中 \mbox 是用来表达数学公式中的普通文本的控制词，其具体用法见 §4.4.4.8 子节。

§4.4.4.5 LaTeX 2ε 数学公式中的二项式系数

在 TeX 源文件中，二项式系数可以用带参数的控制词 \binom{...}{***} 表示，也可以使用控制词 \choose 来表示。但是在 LaTeX 2ε 源文件中，只能用控制词 \choose 来表示二项式系数，具体方法如下：

```
{...\choose ***}
```

其中 ... 和 *** 是数学符号。例如

$$\left\{\begin{matrix}m+n\\ m\end{matrix}\right\}$$

在源文件中的表达为：

```
\[{m+n \choose m}\]
```

注意在上面的表达式中的圆括号是 LaTeX 2ε 自动加上的。

§4.4.4.6 LaTeX 2ε 数学公式中的多行下标

上标和下标是数学公式中最常用的标记，有的时候还会遇到具有多行下标的表达式。例如

$$\sum_{\substack{1\le i\le n\\ 1\le j\le n}} a_{ij}$$

的求和号中的下标有两行，问题是如何输入它。当然我们可以会马上想到用 array 环境来表达上式中的多行下标，即在源文件中用

```
\[
\sum_{
\begin{array}{l}1\le i\le n \\ 1\le j\le n
\end{array}
}
a_{ij}
\]
```

表达上面的公式，但是你会发现这一段源文件对应的的打印输出

$$\sum_{\begin{array}{l}1\le i\le n \\ 1\le j\le n\end{array}} a_{ij}$$

与你想象的不一样。虽然 array 环境中的两行公式也是作为下标放在了求和号的下面，可是它们的尺寸太大了，根本不像下标。为了解决上面的问题，需要使用 TeX 定义的控制词 \atop 。该控制词的作用是把其前面的字符放在其后面的字符之上，具体的使用方法如下：

```
{...\atop ***}
```

其中 ... 为放在上面的字符， *** 为放在下面的字符。例如

$$\begin{matrix}a+b\\ m+n\end{matrix}$$

在源文件中的表达为

```
\[{a+b\atop m+n}\]
```

现在可以在源文件中用

```
\[\sum_{{1\le i\le n}\atop {1\le j\le n}}a_{ij}\]
```

来表示求和号

$$\sum_{\substack{1\le i\le n\\ 1\le j\le n}} a_{ij}$$

§4.4.4.7 LaTeX 2ε 数学公式中的积分号

积分符号是常用的数学符号，在 TeX 中定义的表达积分符号的控制词 \int 和 \oint 同样可以用在 LaTeX 2ε 的源文件中表示积分符号，但是在 $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX 中定义的一些表示重积分号的控制词却不能在 LaTeX 2ε 源文件中使用，例如 \iint 、 \iiint 、 \iiiint 和 \idotsint 等。我们可以用 \int 来表达这些重积分号，下面给出了这些重积分号的一种表示方法：

$\iint$ `$\int\!\!\int$`

$\iiint$ `$\int\!\!\int\!\!\int$`

$\iiiint$ `$\int\!\!\int\!\!\int\!\!\int$`

$\int\cdots\int$ `$\int\cdots\int$`

§4.4.4.8 LaTeX 2ε 数学公式中的文本

有时在数学公式中需要输入一些文本，在 $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX 源文件中可以使用控制词 `\text` 来完成这一任务，但是在 LaTeX 2ε 中却需要用控制词 `\mbox` 来完成数学公式中的文本输入工作，具体方法如下：

```
\mbox{...}
```

其中 ... 代表文本内容。例如在 §4.4.4.4 中的分段函数的表达式中，我们已经用到该结构表达函数分段的条件。

§4.4.4.9 LaTeX 2ε 数学公式中的字体

在默认情况下，LaTeX 2ε 用斜体打印数学公式中的字母，它们代表数学变量。有时我们也需要一些其它的字体表示数学符号，下面给出一些在数学公式中表示不同的字体的控制词。

`\mathrm{...}` `\mathtt{...}` `\mathbf{...}`

`\mathsf{...}` `\mathit{...}` `\mathcal{...}`

其中 ... 表示字母。例如

ABCDEFGHIJKLMNOPQRSTUVWXYZ

ABCDEFGHIJKLMNOPQRSTUVWXYZ

ABCDEFGHIJKLMNOPQRSTUVWXYZ

ABCDEFGHIJKLMNOPQRSTUVWXYZ

ABCDEFGHIJKLMNOPQRSTUVWXYZ

$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

在 LaTeX 2ε 源文件中的表达为：

```
\[
\begin{array}{l}
\mathrm{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
\mathtt{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
```

```
\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\end{array}
\]
```

§4.4.4.10 LaTeX 2ε 数学公式中数学符号的尺寸

单列数学公式符号、正文中的数学公式符号、上下标符号以及更低层上下标符号的尺寸各不相同，例如 \frac 12 在这些公式中的大小分别为：

$$\frac{1}{2} \quad \tfrac{1}{2} \quad \scriptstyle\frac{1}{2} \quad \scriptscriptstyle\frac{1}{2}$$

有时为了使得排版的效果更为美观，需要在源文件中控制数学符号的大小，LaTeX 2ε 提供了如下的命令强制以某种尺寸打印数学符号：

\displaystyle	单列公式尺寸
\textstyle	文本公式尺寸
\scriptstyle	下标尺寸
\scriptscriptstyle	最底层下标尺寸

其中左列是表示公式尺寸的控制词，右列是相对应的含义。事实上，前面给出的分式 $\frac{1}{2}$ 在不同的环境中尺寸的例子在源文件中的表达为

```
$\displaystyle\frac 12$\quad$\frac 12$\quad
$\scriptstyle \frac 12$\quad$\scriptscriptstyle \frac 12$
```

例如连分数

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{a_4+\cfrac{1}{a_5}}}}}$$

在 LaTeX 2ε 源文件中的表达为

```
\[
a_0+\frac 1{\displaystyle a_1
   +\frac 1{\displaystyle a_2
   +\frac 1{\displaystyle a_3
   +\frac 1{\displaystyle a_4
   +\frac 1{a_5}}}}}
\]
```

如果我们去掉上面表达式中的所有控制词 \displaystyle，则相应的打印输出变成

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{a_4+\cfrac{1}{a_5}}}}}$$

这是一个看上去很别扭的连分数表达式。

§4.4.5 在 LaTeX 2ε 中使用 AMS-TeX 的数学符号

美国数学会定义了一些 TeX 的数学字体和符号，在 AMS-TeX 源文件中可以直接使用它们。为了在 LaTeX 2ε 中使用这些字体和符号，需要有宏程序包 amsfonts.sty 和 amssymb.sty，并且在 LaTeX 2ε 源文件中用 §4.2.2 中介绍的方法调入这些宏程序包，即源文件的版面定义部分有如下一行：

```
\usepackage{...,amsfonts,amssymb}
```

其中 ... 代表其它的宏程序包。一旦源文件成功调入宏程序包 amsfonts.sty 和 amssymb.sty，就可以使用在 §3.8 和 §3.9 节中介绍的所有字体与符号。

事实上，在本书的源文件中调入的部分宏程序包有： latexsym.sty、amsfonts.sty 和 amssymb.sty，即在源文件的版面定义部分有如下的一行：

```
\usepackage{latexsym,amsfonts,amssymb}
```

因此可以分别用控制词 \frak 和 \Bbb 打印出 fractur 和空心字体。例如

$$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

$$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

在源文件中的表达方式如下：

```
\[
\begin{array}{l}
\frak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\\
\Bbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\end{array}
\]
```

§4.5 LaTeX 2ε 中的列表

在 LaTeX 2ε 中有三种可用于列表的环境，它们分别是：

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

其中 ... 是列表文本。为了叙述方便，我们分别称这三种环境为 itemize 环境、enumerate 环境和 description 环境。列表文本都要相对于正文的左边界进行缩进对齐，并且有一个标签。标签的具体类型与使用哪一个环境有关，但它们都是用同一控制词 \item 生成。下面分别介绍这几种列表环境。

§4.5.1 LaTeX 2ε 中 itemize 环境列表样例

在 itemize 环境中，列表的标签是一个黑点，即所谓的 bullet。下面一段源文件

```
\begin{itemize}
\item itemize 环境相对于正文的左边界进行缩进对齐，
```

并且有一个标签，该标签是一个黑点，即所谓的 bullet。
\item enumerate 环境的标签是一个从 1 开始的自动编号。
\item description 环境的标签是一个关键词。
\end{itemize}

对应的打印输出为：

- itemize 环境相对于正文的左边界进行缩进对齐，并且有一个标签，该标签是一个黑点，即所谓的 bullet。
- enumerate 环境的标签是一个从 1 开始的自动编号。
- description 环境的标签是一个关键词。

§4.5.2 LaTeX 2ε 中 enumerate 环境列表样例

在 enumerate 环境中，列表的标签是一个从 1 开始的自动编号。下面的一段源文件

\begin{enumerate}
\item itemize 环境相对于正文的左边界进行缩进对齐，
并且有一个标签，该标签是一个黑点，即所谓的 bullet 。
\item enumerate 环境的标签是一个从 1 开始的自动编号。
\item description 环境的标签是一个关键词。
\end{enumerate}

对应的打印输出为：

1. itemize 环境相对于正文的左边界进行缩进对齐，并且有一个标签，该标签是一个黑点，即所谓的 bullet 。
2. enumerate 环境的标签是一个从 1 开始的自动编号。
3. description 环境的标签是一个关键词。

§4.5.3 LaTeX 2ε 中 description 环境列表样例

在 description 环境中，列表的标签是一个关键词。在此环境中，标签的设置按如下的方式：

\item[...]

其中 ... 为表示标签的字词。下面的一段源文件

\begin{description}
\item[圆点] itemize 环境相对于正文的左边界进行缩进对齐，
并且有一个标签，该标签是一个黑点，即所谓的 bullet 。
\item[数字] enumerate 环境的标签是一个从 1 开始的自动编号。
\item[关键词] description 环境的标签是一个关键词。
\end{description}

对应的打印输出为：

圆点 itemize 环境相对于正文的左边界进行缩进对齐，并且有一个标签，该标签是一个黑点，即所谓的 bullet 。

数字 enumerate 环境的标签是一个从 1 开始的自动编号。
关键词 description 环境的标签是一个关键词。

§4.5.4 LaTeX 2ε 中嵌套列表与修改默认标签

本节中给出的列表环境可以相互嵌套，但是嵌套的层数不能超过 4。在 itemize 环境和 enumerate 环境中，随着嵌套层数的不同，标签的样式会自动改变。

在 itemize 环境和 enumerate 环境中我们还可以改变默认的标签形式，具体的命令格式为：

```
\renewcommand{...}{***}
```

其中 ... 为记录指定层次标签样式的命令， *** 为想要修改成的标签形式。下面分别介绍这两个列表环境中标签的具体修改方法。

§4.5.4.1 itemize 标签的修改

在 LaTeX 2ε 的 itemize 环境中，四层的标签分别是由如下的控制词

```
\labelitemi, \labelitemii, \labelitemiii, \labelitemiv
```

所决定。如果想把 itemize 环境的第一层标签改为 * ，可以用如下的命令

```
\renewcommand{\labelitemi}{*}
```

§4.5.4.2 enumerate 标签的修改

在 LaTeX 2ε 的 enumerate 环境中，四层的标签编号分别是由如下的控制词

```
\labelenumi, \labelenumii, \labelenumiii, \labelenumiv
```

所决定，它们可以是 \arabic（阿拉伯数字）、 \roman（小写罗马数字）、 \Roman （大写罗马数字）、 \alph（小写英文字母） 和 \Alph（大写英文字母）。 如果想把 enumerate 环境的第二层标签的编号改为小写罗马数字，使用如下的命令：

```
\renewcommand{\labelenumii}{\roman}
```

在 LaTeX 2ε 的 enumerate 环境中，与四层标签对应的还有四个计数器，它们的名字分别是：

```
enumi, enumii, enumii, enumiv
```

可以用

```
\arabic{...}, \roman{...}, \Roman{...},
\alph{...}, \Alph{...}
```

返回这些计数器的值，这里 ... 代表计数器名。现在可以把第二层标签修改成 1.1 ， 1.2 ， 2.1 ，等的形式，为此仅需使用如下的命令：

```
\renewcommand{\labelenumii}{\arabic{enumi}.\arabic{enumii}}
```

模仿上面的命令格式，可以设计出多种标签编号形式。

§4.6 LaTeX 2ε 中的自动编号与交叉引用

人们喜欢 LaTeX 2ε 的主要原因之一是它的自动编号功能和交叉索引功能，这两个功能使得源文件的编辑者提高了工作效率，特别是提高了修改源文件的效率。例如，如果你的 LaTeX 2ε 源

文件是用第 116 页 §4.3.4 小节中介绍的方法建立了章、节、段标题，那么这些标题将有自动的编号。当你在建立这样的源文件全文后又想要插入一节内容时，只需在源文件的相应的地方继续输入新内容即可， LaTeX 2ε 会为你重新自动安排好各个章、节、段的编号。又如 LaTeX 2ε 的自动编号功能使它可以实现自动建立文章目录的功能，而且这样建立的目录不受排版页面格式（如正文的宽和高等）的改变的影响。实际上我们已经在本章的前面各节中陆续介绍了一些自动编号和交叉引用的方法，本节中只是把这些方法加以总结，提取这种方法的共有特性，以便于读者更好地掌握。

要使用交叉引用功能，必须在引用对象的位置建立一个引用标签，在全文中就是通过这个引用标签来引用该对象。在 LaTeX 2ε 源文件中建立标签的命令如下：

```
\label{...}
```

其中 ... 表示标签名称，它可以使用字母、数字、符号（除了几个特殊字符 ~\#$^_{}% ）的任意组合。注意，在同一个源文件中，不同的对象应该有不同的引用标签，否则会产生错误的引用。

建立了引用标签，可以用如下的几种方式使用该标签来表示它所对应的编号或关键词：

```
\ref{...}
\cite{...}
\pageref{...}
```

其中 ... 为一个已经用 \label 定义的引用标签。这三个引用命令的含义如下：

\ref　产生引用标签对象的自动编号，例如章、节、段的自动编号，数学公式的自动编号，列表的自动编号等。

\cite　产生引用标签所对应参考文献的关键词，并且把该关键词用方括号括起来，具体的例子见 §4.3.7 小节（第 119 页至 121 页）。

\pageref　产生引用标签位置的所在页码。

下面给出一个引用页码的例子。在本书的 LaTeX 2ε 源文件中， §4.6 节的标题为

```
\section{\LaTeXe\  中的列表 \label{4-6}}
```

因此在本书源文件的任何地方都可以用 \pageref{4-6} 表示最后排版 §4.6 节的标题所在页码 139 。另外要注意，有时文章的交叉引用会失败，这时 LaTeX 2ε 会把你引用的编号打印成问号，其原因可能是你在源文件中引用的自动编号的位置位于该引用位置的后面，解决这个问题的方法是连续两次编译源文件. 如果这样还不能解决问题，说明源文件的编写有问题.

§4.7 LaTeX 2ε 中的作图

LaTeX 2ε 中提供了简单的图形功能，这包括画各种点、直线、箭头、圆、 Bézier 曲线段等的功能，本节简单介绍这些图形功能。

§4.7.1 LaTeX 2ε 中的图形环境

LaTeX 2ε 中的图形环境构造如下

```
\begin{picture}(x,y)
...
\end{picture}
```

其中 x 和 y 都是数字，分别代表所构造的图形的宽度和高度值； ... 是一系列画图的命令。

在默认的情况下，图形构造中的宽度和高度以 pt（具体含义见第 9 页第 §1.2.8 小节）为单位。要注意这里的长度和高度的单位由控制词

```
\unitlength
```

所定义。对 \unitlength 赋值的命令如下

```
\setlength{\unitlength}{***}
```

其中 *** 为长度。例如

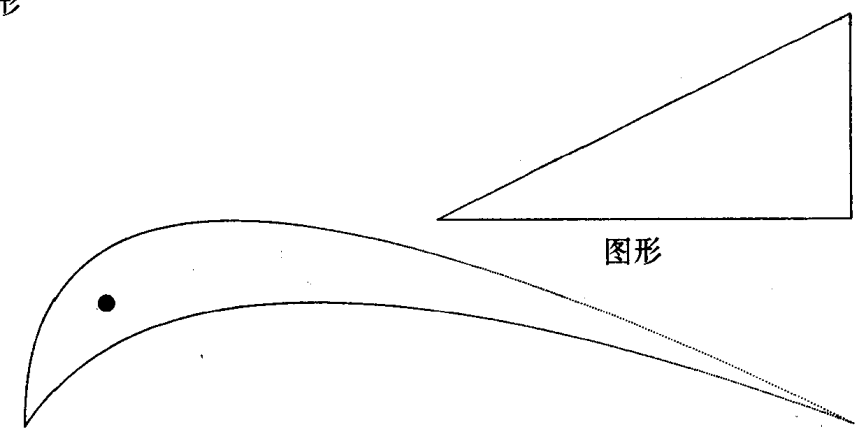
```
\setlength{\unitlength}{.5cm}
```

表示 1 个 \unitlength 长度单位为 0.5cm 。对 \unitlength 赋予新的长度单位后，其起作用的范围是它所在的环境或是直到再对该控制词的重新赋值。

在实际应用的过程中，可以修改 \unitlength 默认的单位，这时构造图形环境的方法如下：

```
\setlength{\unitlength}{***}
\begin{picture}(x,y)
...
\end{picture}
```

例如下面的图形



在源文件中的表达为：

```
\begin{center}
\setlength{\unitlength}{1cm}
\begin{picture}(10,5)
\put(7,2){ 图形 }
\put(1,1.5){\circle*{0.2}}
\qbezier(0,0)(0,5)(10,0)
\qbezier(0,0)(2,3)(10,0)
\put(5,2.5){\line(1,0){5}}
```

```
\put(10,2.5){\line(0,1){2.5}}
\put(10,5){\line(-2,-1){5}}
\end{picture}
\end{center}
```

把该图形环境放在 center 环境中是为了使图形居中安放,此外 center 环境还使得 \unitlength 的新定义仅在该环境内有效。另外这一段源文件中还包括一些绘图命令，我们将会在接下来的几小节中进一步介绍它们。

§4.7.2 LaTeX 2ε 图形环境中的坐标系

在 LaTeX 2ε 的图形环境中，所有的绘图命令都是通过平面坐标系进行。在图形环境中坐标系的原点为图形环境的左下角点，水平方向的坐标轴为 x 轴，竖直方向的坐标轴为 y 轴，整个图形环境都是坐标系的第一象限，坐标轴的长度单位由控制词 \unitlength 所决定。在图形环境中，(x,y) 表示横坐标（即水平坐标）为 x 纵坐标（即竖直坐标）为 y 的点。

§4.7.3 LaTeX 2ε 图形环境中的图形设置命令

在 LaTeX 2ε 中，绘图的功能由图形设置命令来执行。图形设置命令有两种，其一是

```
\put(x,y){...}
```

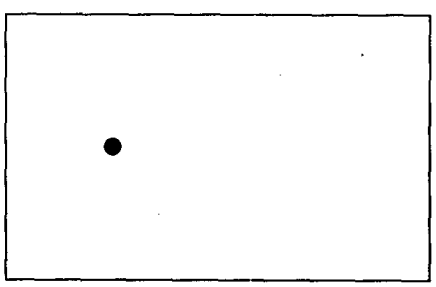
其中 ... 为绘图命令，(x,y) 为绘制的图形安放的位置坐标，它只执行一次花括号中的绘图命令；另外一种图形设置命令的格式如下：

```
\multiput(x,y)(dx,dy){n}{...}
```

其中 ... 为绘图命令，(x,y) 为绘制的图形安放的初始位置坐标，(dx,dy) 为增量坐标，n 是花括号中的图形命令重复执行的次数，每次重复执行花括号中的绘图命令，图形安放的初始位置坐标都增加 (dx,dy) 。为了区别起见，我们称 \put 为单图形设置命令，称 \multiput 为多图形设置命令。

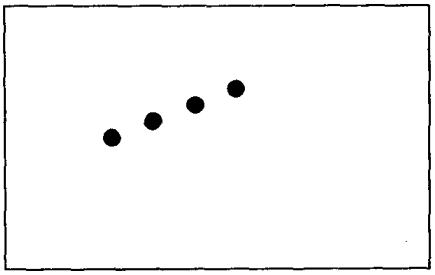
下面给出使用单图形设置命令的例子，其左边是一段 LaTeX 2ε 源文件，右边的方框的内部是该段源文件对应的打印输出。在这个例子中，只是在图形环境中的 (1,1.5) 坐标处画了一个实心圆。

```
\setlength{\unitlength}{1cm}
\begin{picture}(5,3)
\put(1,1.5){\circle*{0.2}}
\end{picture}
```



下面给出使用多图形设置命令的例子，其左边是一段 LaTeX 2ε 源文件，右边的方框的内部是该段源文件对应的打印输出。在这个例子中，\multiput 的初始位置坐标为 $(1,1.5)$ ，增量坐标是 $(0.5,0.2)$ ，重复了四次画实心圆的命令。由初始坐标和增量坐标可以算得：第一个圆的中心坐标为 $(1,1.5)$ ，第二个圆的中心坐标为 $(1,1.5)+(0.5,0.2)=(1.5,1.7)$ ，第三个圆的中心坐标为 $(1.5,1.7)+(0.5,0.2)=(2,1.9)$ ，第四个圆的中心坐标为 $(2,1.9)+(0.5,0.2)=(2.5,2.1)$ 。

```
\setlength{\unitlength}{1cm}
\begin{picture}(5,3)
\multiput(1,1.5)(0.5,0.2){4}{\circle*{0.2}}
\end{picture}
```



§4.7.4 LaTeX 2ε 图形环境中的基本绘图命令

本小节介绍图形设置命令中需要使用的基本绘图命令。

§4.7.4.1 LaTeX 2ε 图形环境中的文本

在 LaTeX 2ε 图形环境中输出文本是很简单的一件事，只需在 §4.7.3 中介绍的图形设置命令里输入绘图命令的地方输入文本即可。例如

```
\setlength{\unitlength}{1cm}
\begin{picture}(5,3)
\put(0,2.5){ 单图形设置命令 }
\multiput(0,0)(0.5,0.5){4}{ 复 图
形设置命令 }
\end{picture}
```

单图形设置命令

复图形设置命令
复图形设置命令
复图形设置命令
复图形设置命令

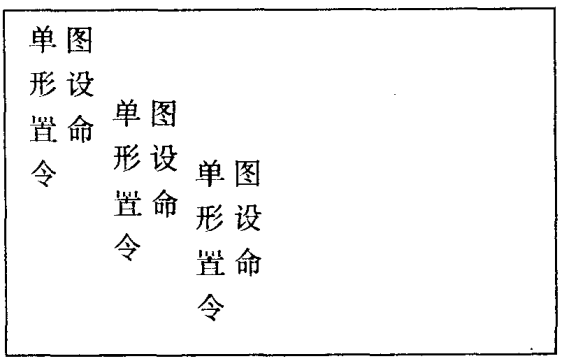
在上面的例子中，使用了 \put 命令在图形环境中输出文本“单图形设置命令”，该文本的左下角坐标恰为该命令的位置坐标 $(0,2.5)$ 。例子中还使用了 \multiput 命令在图形环境中重复四次输出文本“复图形设置命令”，这些文本的左下角坐标分别为该命令决定的位置坐标 $(0,0)$ 、$(0.5,0.5)$ 、 $(1,1)$ 和 $(1.5,1.5)$ 。

另外还可以在输出的文本中使用 \parbox 输入多行文本，具体有如下的三种方式：

\put(x,y){\parbox[b]{***}{...}}	多行文本的宽度为 *** ，文本内容为 ... ，文本左下角的坐标为 (x,y) 。
\put(x,y){\parbox{***}{...}}	多行文本的宽度为 *** ，文本内容为 ... ，文本左边中心位置坐标为 (x,y) 。
\put(x,y){\parbox[t]{***}{...}}	多行文本的宽度为 *** ，文本内容为 ... ，文本左边与第一行文本的底线的交点坐标为 (x,y) 。

下面的例子显示了输入多行文本的三种方法的特点。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(0,2){\parbox[b]{.8cm}{ 单 图
形设置命令 }}
\put(1,2){\parbox{.8cm}{ 单 图 形
设置命令 }}
\put(2,2){\parbox[t]{.8cm}{ 单 图
形设置命令 }}
\end{picture}
```



§4.7.4.2 LaTeX 2ε 图形环境中的直线

LaTeX 2ε 图形环境中绘线的命令如下：

```
\line(dx,xy){...}
```

其中 ... 是线段的长度值，(dx,dy) 为直线的方向向量。这里的 dx 和 dy 的取值必须遵循如下的规则：

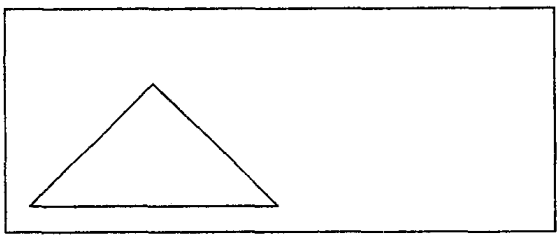
1. 数值必须是整数（正负均可）
2. 整数的绝对值不能超过 6
3. dx 和 dy 不能有公因子

对于不同的方向向量，线段的长度值有不同的含义，具体区别如下：

- 当方向向量 $(dx,dy)=(1,0)$ 、 $(dx,dy)=(-1,0)$ 、 $(dx,dy)=(0,1)$ 或 $(dx,dy)=(0,-1)$ 时，线段的长度值是绘出线段的实际长度。
- 对于其它方向向量，线段的长度值是绘出线段在水平方向投影的长度。

下面是绘制直线的例子：

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,2.5)
\put(0,0.2){\line(1,0){3}}
\put(0,0.2){\line(1,1){1.5}}
\put(3,0.2){\line(-1,1){1.5}}
\end{picture}
```



其中第一条绘线命令是从 (0,0.2) 开始画一条长度为 3 的水平线段；第二条绘线命令是从 (0,0.2) 开始沿 (1,1) 方向（即与水平方向成 45° 角方向）画一条水平投影长度为 1.5 的线段；第三条绘线命令是从 (3,0.2) 开始沿 (−1,1) 方向（即与水平方向成 135° 角方向）画一条水平投影长度为 1.5 的线段。上面的三条线段恰好构成一个三角形。

注意，在 LaTeX 2ε 的标准配置中我们只能画出 25 种斜率的直线，并且直线的长度不能短于 10pt 或 3.5mm 。要摆脱这种限制，需要包含特殊的宏程序包（如 pict2e.sty ），或者用第 152 页 §4.7.4.13 子节中介绍的 emtex 的画线命令。

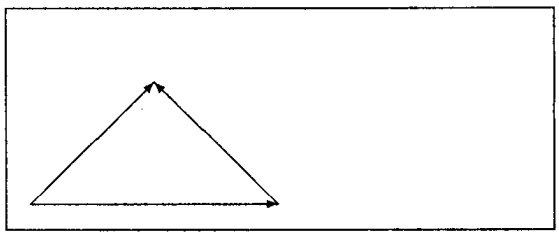
§4.7.4.3 LaTeX 2ε 图形环境中的箭头

LaTeX 2ε 图形环境中绘箭头的命令如下：

```
\vector(dx,xy){...}
```

其中 ... 是线段的长度值，(dx,dy) 为箭头的方向向量。绘制箭头命令的使用方法与 §4.7.4.2 子节中绘制直线命令的使用方法基本相同，所不同的是 dx 和 dy 取值的范围更窄。在这里仅给出一个例子。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,2.5)
\put(0,0.2){\vector(1,0){3}}
\put(0,0.2){\vector(1,1){1.5}}
\put(3,0.2){\vector(-1,1){1.5}}
\end{picture}
```



上面这个例子仅是把 §4.7.4.2 子节的例子里的 \line 换成了 \vector 。

§4.7.4.4 LaTeX 2ε 图形环境中的矩形

在 LaTeX 2ε 图形环境中，有三种绘制矩形的方法，它们分别是：

```
\makebox(w,h)[***]{...}
\framebox(w,h)[***]{...}
\dashbox{l}(w,h)[***]{...}
```

其中 w 和 h 分别为矩形的宽度和高度值，*** 为矩形的内文本对齐参数，... 为在矩形中放置的文本，l 为虚线尺寸参数。在矩形的绘制结果中，其左下角的坐标等于放置命令 \put 的位置坐标。

三个绘制矩形的区别在于矩形边框的不同，\makebox 没有边框，\framebox 的边框为实线，而 \dashbox 的边框为虚线。另外，只有 \dashbox 中有虚线尺寸参数，它决定了组成虚线的一个小线段的长度。虽然 \makebox 画不出矩形，但是它有一些特殊的用途，比如说在图形中精确定位文本等。

下面简单列出文本对齐参数可以取的值及含义，其中左边列出的是参数可以取的值，右边是该参数值对齐方式的解释。

- t 输入的文本水平居中地位于顶边框的下边，这里的 t 是英文单词 top 的第一个字母。
- b 输入的文本水平居中地位于底边框的上边，这里的 b 是英文单词 bottom 的第一个字母。
- l 输入的文本垂直居中地位于左边框的右边，这里的 l 是英文单词 left 的第一个字母。
- r 输入的文本垂直居中地位于右边框的左边，这里的 r 是英文单词 right 的第一个字母。
- s 输入的文本垂直居中，但是文本的宽度要水平伸展以充满整个矩形的宽度。
- tl 输入的文本位于矩形的左上角，t 和 l 分别是英文单词 top 和 left 的第一个字母。另外该选项还可以写成 lt 。

tr 输入的文本位于矩形的右上角，t 和 r 分别是英文单词 top 和 right 的第一个字母。另外该选项还可以写成 rt 。

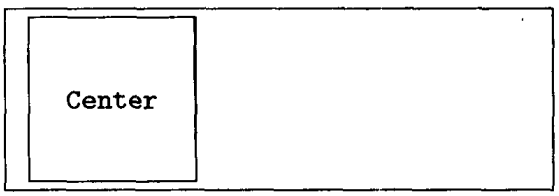
bl 输入的文本位于矩形的左下角，b 和 r 分别是英文单词 bottom 和 left 的第一个字母。另外该选项还可以写成 lb 。

br 输入的文本位于矩形的右下角，b 和 r 分别是英文单词 bottom 和 right 的第一个字母。另外该选项还可以写成 rb 。

另外在省略文本对齐参数时，文本被置于矩形的中心位置。

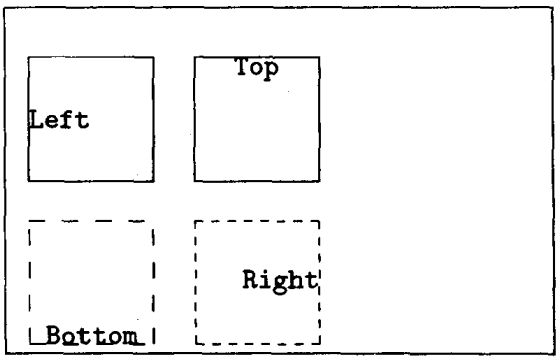
下面给出一些绘制矩形的例子，读者对照左边源文件和其右边的图形输出可以进一步掌握各个参数的含义。第一个是省略文本对齐参数的例：

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
\put(0,0){\framebox(2,2){Center}}
\end{picture}
```



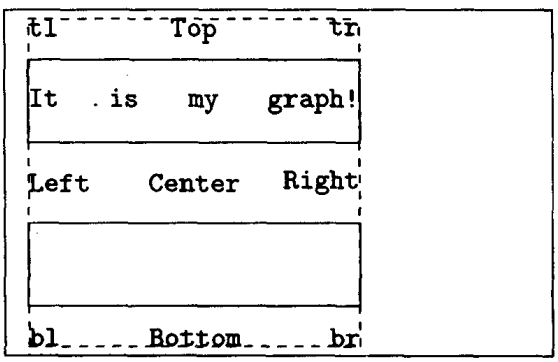
下面是实边矩形与虚边矩形的例子。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(0,2){\framebox(1.5,1.5)[l]{Left}}
\put(2,2){\framebox(1.5,1.5)[t]{Top}}
\put(0,0){\dashbox{0.2}(1.5,1.5)[b]{Bottom}}
\put(2,0){\dashbox{0.1}(1.5,1.5)[r]{Right}}
\end{picture}
```



下面是无边矩形、虚边矩形以及各种文本对齐参数的例子。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(0,0){\dashbox{0.1}(4,4)[l]{Left}}
\put(0,0){\makebox(4,4)[r]{Right}}
\put(0,0){\makebox(4,4)[t]{Top}}
\put(0,0){\makebox(4,4)[b]{Bottom}}
\put(0,0){\makebox(4,4){Center}}
\put(0,0){\makebox(4,4)[tl]{tl}}
\put(0,0){\makebox(4,4)[tr]{tr}}
\put(0,0){\makebox(4,4)[bl]{bl}}
\put(0,0){\makebox(4,4)[br]{br}}
\put(0,2.5){\framebox(4,1)[s]{It is my graph!}}
\put(0,0.5){\framebox(4,1){}}
\end{picture}
```



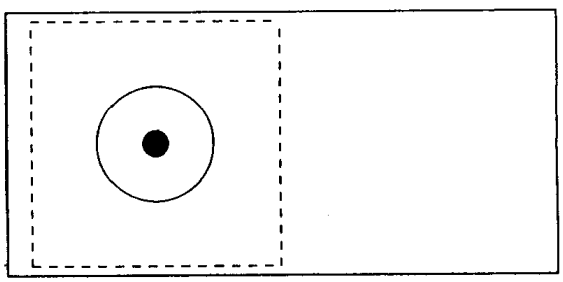
§4.7.4.5 LaTeX 2ε 图形环境中的圆

在 LaTeX 2ε 图形环境中，绘制空心圆和实心圆的命令分别是

\circle{d} 和 \circle*{d}

其中 d 为圆的直径。用这两个命令绘制出圆的圆心坐标等于放置命令 \put 的位置坐标。下面的例子演示了绘制圆的命令：

```
\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
\put(0,0){\dashbox{0.1}(3,3){}}
\put(1.5,1.5){\circle{2}}
\put(1.5,1.5){\circle{1.5}}
\put(1.5,1.5){\circle*{.3}}
\end{picture}
```



在上面的例子中，虽然前两条画圆命令的直径分别是 2 和 1.5，但是 LaTeX 2ε 实际却只画出一个圆，其原因是在 LaTeX 2ε 的标准配置中只能画出几个固定尺寸的圆。对于给定的直径参数，LaTeX 2ε 会画出一个最匹配给定直径的圆。

§4.7.4.6 LaTeX 2ε 图形环境中的圆角矩形

在 LaTeX 2ε 中，绘制圆角矩形的命令为：

\oval(w,h)[...]

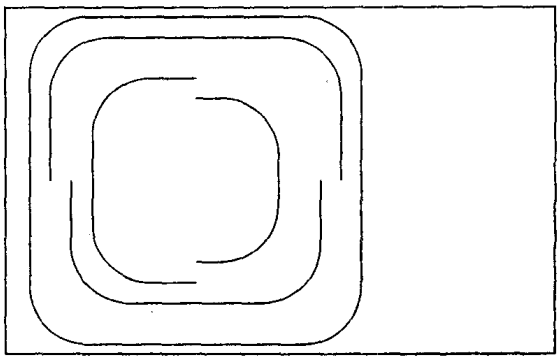
其中 w 为圆角矩形的宽度值，h 为圆角矩形的高度值，... 为部分参数。用这个命令绘制出圆角矩形的中心坐标等于放置命令 \put 的位置坐标。下面列出部分参数 ... 的可选值及含义：

- t 只画圆角矩形的上面一半，这里的 t 是英文单词 top 的第一个字母。
- b 只画圆角矩形的下面一半，这里的 b 是英文单词 bottom 的第一个字母。
- l 只画圆角矩形的左面一半，这里的 l 是英文单词 left 的第一个字母。
- r 只画圆角矩形的右面一半，这里的 r 是英文单词 right 的第一个字母。
- tl 只画圆角矩形的左上面的四分之一，这里的 t 和 l 分别是英文单词 top 和 left 的第一个字母。该选项可以等价地写成 lt 。
- tr 只画圆角矩形的右上面的四分之一，这里的 t 和 r 分别是英文单词 top 和 right 的第一个字母。该选项可以等价地写成 rt 。
- bl 只画圆角矩形的左下面的四分之一，这里的 b 和 l 分别是英文单词 bottom 和 left 的第一个字母。该选项可以等价地写成 lb 。
- br 只画圆角矩形的右下面的四分之一，这里的 b 和 l 分别是英文单词 bottom 和 right 的第一个字母。该选项可以等价地写成 rb 。

当省略部分参数时，LaTeX 2ε 画出整个圆角矩形。

下面的例子演示了绘制圆角矩形命令的使用方法。

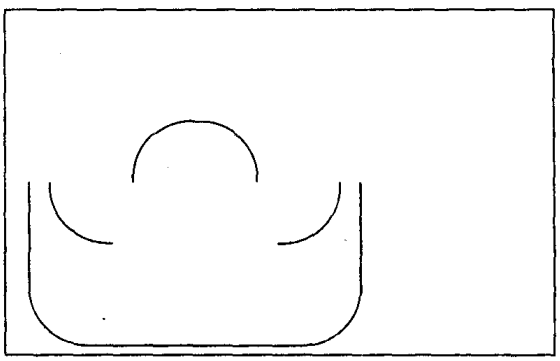
```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(2,2){\oval(4,4)}
\put(2,2){\oval(3.5,3.5)[t]}
\put(2,2){\oval(3,3)[b]}
\put(2,2){\oval(2.5,2.5)[l]}
\put(2,2){\oval(2,2)[r]}
\end{picture}
```



其中第一条绘图命令画出了一个长和宽都为 4cm 的圆角矩形，该矩形的中心点坐标为 (2,2)；第二条绘图命令画出了一个长和宽都为 3.5cm 的圆角矩形的上半部分，该矩形的中心点坐标为 (2,2)；第三条绘图命令画出了一个长和宽都为 3cm 的圆角矩形的下半部分，该矩形的中心点坐标为 (2,2)；第四条绘图命令画出了一个长和宽都为 2.5cm 的圆角矩形的左半部分，该矩形的中心点坐标为 (2,2)；第五条绘图命令画出了一个长和宽都为 2cm 的圆角矩形的右半部分，该矩形的中心点坐标为 (2,2)。

利用 \oval 命令，我们还可以画出特定尺寸的半圆和四分之一圆，下面的例子显示了具体的画法。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(2,2){\oval(4,4)[b]}
\put(2,2){\oval(1.5,1.5)[t]}
\put(1,2){\oval(1.5,1.5)[bl]}
\put(3,2){\oval(1.5,1.5)[br]}
\end{picture}
```



§4.7.4.7 LaTeX 2ε 图形环境中的竖直文本

有时我们想要在图形环境中按竖直排列的方式打印一个单词，这可以通过如下的命令来完成：

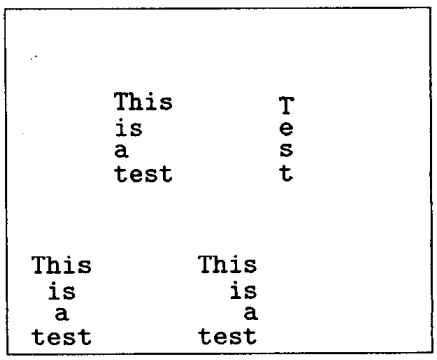
```
\shortstack[...]{***}
```

其中可选参数 ... 为列对齐参数，*** 为要输入的文本，文本的行与行之间用 \\ 分开。用这个命令绘制出文本框的左下角的坐标等于放置命令 \put 的位置坐标。当缺省对齐参数时，LaTeX 2ε 取列中心对齐的方式。列对齐参数可以取的值列举如下：

- l 表示左对齐。
- r 表示右对齐。
- c 表示中心对齐。

下面给出使用 \shortstack 在图形环境中打印竖直文本的例子。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(0,0){\shortstack{This\\ is\\ a\\ test}}
\put(1,2){\shortstack[l]{This\\ is\\ a\\ test}}
\put(2,0){\shortstack[r]{This\\ is\\ a\\ test}}
\put(3,2){\shortstack{T\\ e\\ s\\ t}}
\end{picture}
```



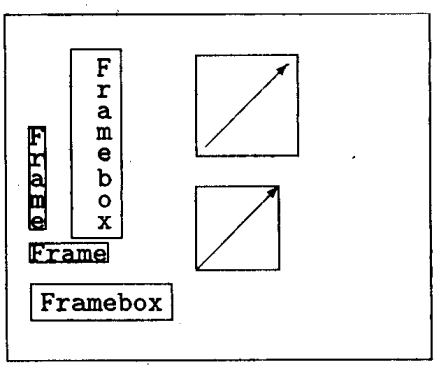
§4.7.4.8 LaTeX 2ε 图形环境中的有框文本

虽然在 §4.7.4.4 子节中的绘制矩形的命令可以把文本放在一个矩形框中，但是这个矩形框要比输出文本所占的面积大一些，使得文本和边框之间出现一些多余的空白，特别是当方框中包含有图形指令时更是如此。为了使边框尽量靠近它所包含的内容，可以使用如下的命令：

```
\frame{...}
```

其中 ... 是文本内容。下面的例子演示了 \frambox 和 \frame 之间的区别。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(0,0.5){\framebox{Framebox}}
\put(0,1.1){\frame{Frame}}
\put(0,1.5){\frame{\shortstack{F\\r\\a\\m\\e}}}
\put(0.5,1.5){\framebox{
\shortstack{F\\r\\a\\m\\e\\b\\o\\x}}}
\put(2,1){\frame{\vector(1,1){1}}}
\put(2,2.5){\framebox{\vector(1,1){1}}}
\end{picture}
```



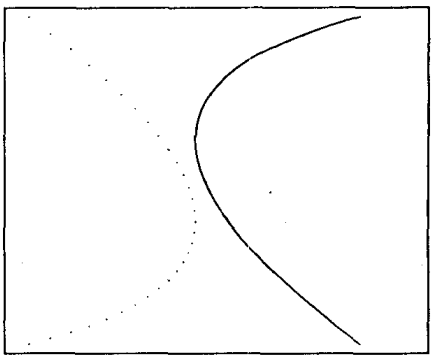
§4.7.4.9 LaTeX 2ε 图形环境中的 Bézier 曲线

人们常用 Bézier 曲线段来拟合复杂的曲线，在 LaTeX 2ε 的图形环境中，提供了绘制 Bézier 曲线的功能，具体可以用如下的两个命令来实现：

```
\bezier{n}(x1,y1)(x2,y2)(x3,y3)
\qbezier[n](x1,y1)(x2,y2)(x3,y3)
```

其中 (x_1,y_1) 、(x_2,y_2) 和 (x_3,y_3) 分别是 Bézier 曲线的三个控制点的坐标， n 是一个正整数，它表示实际画该曲线所用的点数减一。上面的两个命令都是画一条从 (x_1,y_1) 到 (x_3,y_3) 的 Bézier 曲线，该曲线在 (x_1,y_1) 点的切线段为从 (x_1,y_1) 到 (x_2,y_2) 的直线段，该曲线在 (x_3,y_3) 点的切线段为从 (x_3,y_3) 到 (x_2,y_2) 的直线段。两个命令的区别是 \qbezier 中的 n 是可以省略的参数，当省略它时，LaTeX 2ε 就绘制出一条光滑的 Bézier 曲线。另外，这两条命令也可以不借助于置放命令 \put 或 \multiput 而单独使用。下面例子演示了这两条命令的效果。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\bezier{30}(0,0)(4,1)(0,4)
\qbezier(4,0)(0,3)(4,4)
\end{picture}
```



在上面的例子中，第一条 Bézier 曲线是由 31 个点画出来的，而第二条 Bézier 曲线是光滑曲线。

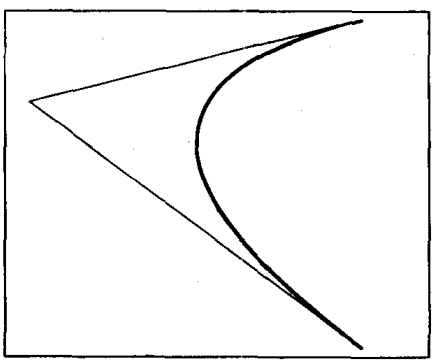
§4.7.4.10 LaTeX 2ε 图形环境中线的粗细

在 LaTeX 2ε 的图形环境中，存在着两种可供选择的线宽。可以用控制词

\thicklines 和 \thinlines

来表示选择粗线或是细线来画图。这两个命令作用范围是它们所在的图形环境或者是在遇到相反的线宽选择命令之前。图形环境的默认线宽是细线。下面的例子演示了这两个命令的使用方法。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(4,0){\line(-4,3){4}}
\thicklines
\qbezier(4,0)(0,3)(4,4)
\thinlines
\put(4,4){\line(-4,-1){4}}
\end{picture}
```

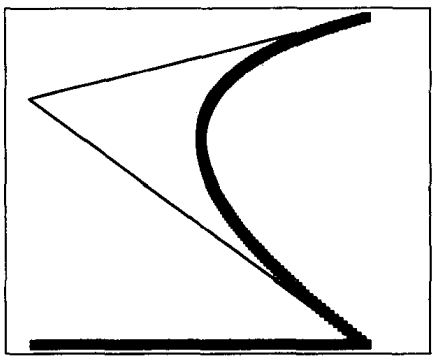


我们还可以修改粗线的宽度，其方法如下：

\linethickness{...}

其中 ... 为长度。例如 \linethickness{1.5mm} 表示把粗线的宽度定义为 1.5mm 。需要注意的是斜线的宽度不受 \linethickness 的影响，下面的例子演示了 \linethickness 的作用效果。

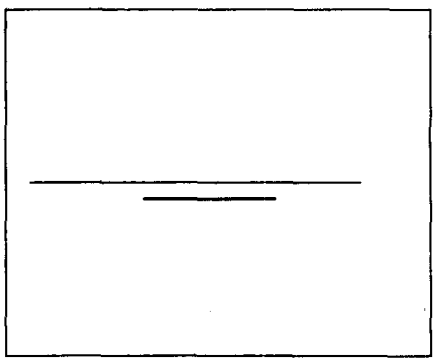
```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\thicklines
\put(4,0){\line(-4,3){4}}
\linethickness{1.2mm}
\qbezier(4,0)(0,3)(4,4)
\put(4,4){\line(-4,-1){4}}
\put(0,0){\line(1,0){4}}
\end{picture}
```



§4.7.4.11 LaTeX 2ε 图形环境中嵌套图形

在 LaTeX 2ε 的图形环境的设置命令 \put 和 \multiput 中，还可以使用另外的图形环境，即允许图形环境的嵌套使用。为了叙述方便，我们称里层的图形环境为子图形环境，外层的图形环境为父图形环境。下面的例子演示了图形环境的嵌套使用。

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,4)
\put(1,1)
    {
    \setlength{\unitlength}{0.4cm}
    \begin{picture}(4,4)
    \thicklines
    \put(0,2){\line(1,0){4}}
    \end{picture}
    }
\put(0,2){\line(1,0){4}}
\end{picture}
```



注意在上面的嵌套图形环境中，同一个绘图的命令 \put(0,2){\line(1,0){4}} 在子图形环境中和父图形环境中画出线的起始点不一样，长度也不一样。原因一是在子环境中的绘图命令的坐标是相应于子环境的坐标系，该坐标系的原点恰为其父图形环境坐标系的 (1,1) 点，所以同样的绘制线的命令在子图形环境中和父图形环境中会有不同的起点。原因二是子图形环境坐标长度单位为 0.4cm，因此在子图形环境中 \put(0,2){\line(1,0){4}} 画出线的实际长度为 $4 \times 0.4 = 1.6$cm，而不是 4cm。另外，在子图形环境中的命令 \thicklines 仅在子环境中有效，因此子环境中的线是粗线，父环境中的线是细线。

§4.7.4.12 LaTeX 2ε 中部分图形的储存与调用

如果文章中大量使用同样的图形结构，可以把这些结构以子图的形式定义成一个控制词，然后用该控制词来表达这个结构。定义表示子图的控制词用如下的方法：

```
\newsavebox{~~~}
 \savebox{~~~}(w,h)[***]{...}
```

其中 ~~~ 为控制词的名称； w 为子图的宽度， h 为子图的高度； *** 为子图内的对齐参数，其含义与 §4.7.4.4 子节中 \makebox 的位置对齐参数完全相同； ... 为子图的具体内容。调用已经定义的子图控制词的方法如下

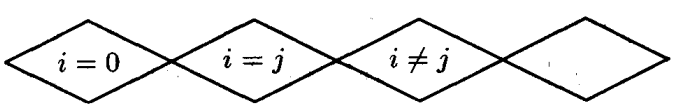
```
\usebox{~~~}
```

其中 ~~~ 为表示子图的控制词。

下面的例子显示如何应用上面的方法定义和调用子图。

```
\begin{center}
\setlength{\unitlength}{1cm}
\newsavebox{\mytest}
\savebox{\mytest}(0,0)
{
\thicklines
\put(0,0.5){\line(-2,-1){1}}
\put(0,0.5){\line(2,-1){1}}
\put(0,-0.5){\line(-2,1){1}}
\put(0,-0.5){\line(2,1){1}}
}
\begin{picture}(10,3)
\put(1,1){\usebox{\mytest}\makebox(0,0){$i=0$}}
\put(3,1){\usebox{\mytest}\makebox(0,0){$i=j$}}
\put(5,1){\usebox{\mytest}\makebox(0,0){$i\ne j$}}
\put(7,1){\usebox{\mytest}\makebox(0,0)}
\end{picture}
\end{center}
```

在这段源文件中，首先定义了一个表示菱形的控制词 \mytest ，然后在图形环境中四次调用该子图。在定义控制词 \mytest 时，把子图的长和宽都定义成 0 ，为的是以后用 \put 命令调用该子图时，使得 \put 安置坐标的位置恰好等于子图的中心点。正是因为这样，才能使得我们在调用该子图时可以在菱形的中心写入文本。



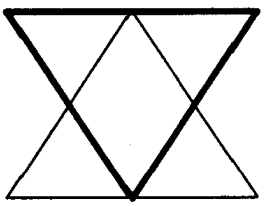
§4.7.4.13 emtex 中的画线命令

在 §4.7.4.2 子节中介绍过 LaTeX 2ε 的标准程序包或提供的画线命令只能画出 25 种特定斜率的直线，而且线的宽度不能随意改变。为了摆脱这种限制可以使用 emtex 软件包中提供的

画线命令。下面列出软件包 emtex 提供的一些常用画线命令。

\special{em:linewidth w}	其中 w 为长度。该命令将缺省线宽的宽度设置为 w 宽，缺省的线宽为 0.4pt 。
\special{em:moveto}	将当前的位置设为画线的当前点。
\special{em:lineto}	画一条由当前点到当前位置的直线，然后把当前位置设置为当前点。

下面以例子说明上面几个命令的用法。考察下面的由两个三角形构成的图形：



其在源文件中的表达如下：

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,2.5)
\put(0,0.2){\special{em:moveto}}
\put(3,0.2){\special{em:lineto}}
\put(1.5,2.5){\special{em:lineto}}
\put(0,0.2){\special{em:lineto}}
\special{em:linewidth 2pt}
\put(0,2.5){\special{em:moveto}}
\put(3,2.5){\special{em:lineto}}
\put(1.5,0.2){\special{em:lineto}}
\put(0,2.5){\special{em:lineto}}
\end{picture}
```

其中由细线构成的三角形是前四个设置命令 \put 的结果；接下来的命令把线形的宽度设置成为 2pt ，使得以后的线形变粗；最后的四个设置命令 \put 产生了粗线构成的三角形。

§4.8 LaTeX 2ε 画图辅助软件 texcad 简介

由于在编写 LaTeX 2ε 的图形源文件时不能立刻看见绘图命令所对应图形输出效果，使得对图形的编写和修改效率大为降低。为了克服这种不足， Georg Horn 编写了 LaTeX 画图辅助软件 texcad ，该软件在 Dos 环境下运行，提供了所见即所得的画图环境。在该环境中，可以通过鼠标和键盘绘出 §4.7.4 小节中介绍的各种图形，并且可以把最后的绘图结果用 LaTeX 2ε 的图形语言保存在一个扩展名为 .pic 的文件中。在 LaTeX 2ε 源文件中，可以用 \input 命令把这种图形文件调入到源文件中恰当的位置。本节简单介绍 texcad 的使用方法。

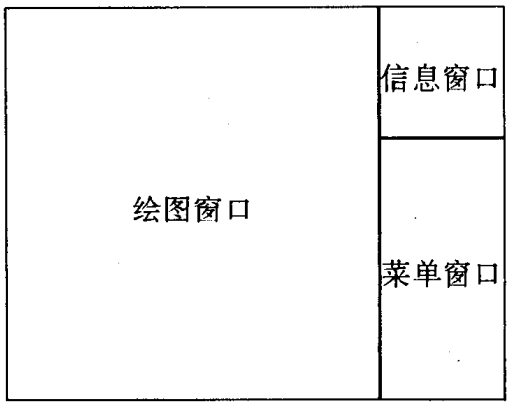
§4.8.1 texcad 软件的运行与主菜单窗口

texcad 为免费软件，在该软件中包含一个名为 tcman.dvi 文件，它是软件的使用手册，详细介绍了该软件的安装和使用方法。利用 TeX 的屏幕显示或打印机驱动程序，可以把该使用手册显示或打印出来。在本小节中只是介绍该软件的运行界面。

假设已经安装好了 texcad 软件，可以在 dos 命令行上键入：

texcad ↩

此时会出现该软件的界面，它的构造如下图所示：



其中绘图窗口是用鼠标绘图或者用键盘输入图形文本的窗口，也是显示所画出图形效果的窗口；信息窗口是程序给出的各种信息提示窗口；菜单窗口是选择各种绘图功能的窗口。 texcad 窗口中的各种命令主要通过鼠标来进行，例如在菜单窗口中选取绘图功能和在绘图窗口中进行绘图等。另外，键盘右上角的 Esc 也是一个非常重要的键，它的主要功能是返回到上一级菜单，当然鼠标右键也具有此功能。在该软件的命令执行中，可以把鼠标的左键理解为 " yes "，把鼠标的右键理解为 " no "。

下面以 3.0 版的 texcad 为例介绍各个窗口的使用方法。菜单窗口中主菜单选项和相应的含义列举如下：

MAIN-MENU	主菜单标识
Draw	绘图命令菜单
Edit	图形编辑菜单
Redraw	刷新绘图窗口中的图形
Snap	鼠标位移量选择菜单
Zoom	屏幕显示比例缩放菜单
Window	绘图窗口内容平移菜单
Options	软件运行参数选择菜单
Save	图形存盘
Load	载入存盘图形
Quit	退出该程序

下面只介绍绘图命令菜单、图形编辑菜单、屏幕显示比例缩放菜单、绘图窗口内容平移菜单、图形存盘选项以及载入存盘图形选项，感兴趣其它菜单选项的读者可以进一步查阅该软件

的使用手册。

§4.8.2 texcad 软件的绘图命令菜单及绘图

当用鼠标的左键点击主菜单窗口中的 Draw 时，就会进入绘图状态。此时菜单窗口出现绘图命令菜单，其内容及含义如下：

DRAW	绘图命令菜单标识
Text	输入文本
Framebox	绘制实线矩形框
Dashbox	绘制虚线矩形框
Filled Box	绘制填充矩形
Line	绘制直线
Vect	绘制箭头直线
Circle	绘制空心圆
Filled Circle	绘制实心圆
Oval	绘制圆角矩形
Half/Quarter Oval	绘制 1/2 或 1/4 圆角矩形
Bezier	绘制 Bézier 曲线

下面介绍几个具体的菜单选项，其它的菜单选项的使用过程类似。

§4.8.2.1 绘图命令菜单的 Text 选项与文本输入

本子节介绍 Text 选项，用鼠标点击该选项，信息窗口将会出现如下的信息：

Reference-point:

该信息提示使用者选择所输入文本的位置。现在移动鼠标，将会出现一个定位十字线，它的中心将会随鼠标移动而移动。将十字线的中心移动到想要输入文本的位置后，按动鼠标左键，表示此时十字的中心将作为输入文本的参考点。按动鼠标左键后，屏幕的左上角会出现如下的信息

text:

该行的下面是一个表示需要键盘输入的光标。现在可以用键盘输入文本，输入完文本后按回车键或是按鼠标左键，表示完成文本输入。此时菜单选择窗口将出现所输入文本相对于参考点的水平定位方式的选项，可以选择的选项和相应的含义如下：

Left:	输入文本的左端与参考点的横坐标对齐
Right:	输入文本的右端与参考点的横坐标对齐
Center:	输入文本的水平中心位置与参考点的横坐标对齐

在所需要的选项上点击鼠标的左键，表示选定该选项。此时菜单选择窗口将出现选择所输入文本相对于参考点的垂直定位方式的选项，可以选择的选项和相应的含义如下：

Top:	输入文本的上端与参考点的纵坐标对齐
Bottom:	输入文本的下端与参考点的纵坐标对齐
Center:	输入文本的垂直中心位置与参考点的纵坐标对齐

在所需要的选项上点击鼠标的左键，表示选定该选项。此时已完成文本的输入，程序将在该文本的参考点处显示一个大写字母 T ，表示该处有一个文本；而菜单窗口将重新出现绘图命令菜单 DRAW 。

§4.8.2.2 绘图命令菜单的 Framebox 选项与矩形绘制

现在介绍 Framebox 选项。用鼠标左键点击该选项，信息窗口将出现如下的信息：

```
Lower left corner:
```

表示要使用者选择矩形的左下角坐标。此时可以用鼠标移动定位十字的中心，在合适的位置按动鼠标的左键。此时信息窗口中将会出现如下的信息：

```
Upper right corner:
```

表示要使用者选择矩形的右上角坐标。此时可以用鼠标移动定位十字的中心，在合适的位置按动鼠标的左键。此时屏幕的左上角将会出现如下的信息：

```
BoxText:
```

表示要使用者决定是否要在矩形的内部输入文本。如果不想输入文本，可以按回车键或是点击鼠标左键；如果想要输入文本，则可以用键盘输入文本，其步骤与 §4.8.2.1 子节中类似。

§4.8.2.3 绘图命令菜单的 Line 选项与直线的绘制

下面介绍 Line 选项。用鼠标左键点击该选项，信息窗口将出现如下的信息：

```
First point:
```

表示要使用者选择直线的起点坐标。此时可以用鼠标移动定位十字的中心，在合适的位置按动鼠标的左键。此时信息窗口将会出现如下的信息：

```
Second point:
```

在合适的位置按动鼠标的左键，此时在信息窗口中还会出现如下的信息：

```
Second point:
```

表示要使用者决定是否要继续画折线。如果不想画折线，可以点击鼠标右键；如果想画折线，则可以继续用鼠标选择折线的下一点。

§4.8.2.4 绘图命令菜单的 Circle 选项与空心圆的绘制

下面介绍 Circle 选项。用鼠标左键点击该选项，信息窗口将出现如下的信息：

```
Center-point:
```

表示要使用者选择圆心坐标。此时可以用鼠标移动定位十字的中心，在合适的位置按动鼠标的左键。此时信息窗口将增加一条新信息：

```
Center-point:     Radius:
```

表示要使用者选择圆的半径。移动鼠标，绘图窗口中将出现一个新的空心圆，它的半径会随鼠标的移动而在一定范围内变化。当圆的大小合适时按动鼠标左键，就完成了此空心圆的绘制。

§4.8.2.5 绘图命令菜单的 Bezier 选项与 Bézier 曲线的绘制

下面介绍 Bezier 选项。用鼠标左键点击该选项，信息窗口将出现如下的信息：

```
First point:
```

表示要使用者选择 Bézier 曲线的起始坐标。可以用鼠标移动定位十字的中心，在合适的位置按动鼠标的左键选择曲线的起始点坐标。现在信息窗口将增加一条新信息：

```
Second point:
```

表示要使用者选择曲线的结束点坐标。移动鼠标，在合适的位置按动鼠标的左键选择合适的曲线结束点坐标。现在信息窗口又将增加一条新信息：

```
Third point
```

表示要使用者选择曲线控制点（控制曲线两端点切线的点）的坐标。移动鼠标，屏幕上就会出现一条新的曲线，其形状会随鼠标的移动而变化。在合适的位置定位十字线中心，按动鼠标的左键就完成这条 Bézier 曲线的绘制工作。

§4.8.3 texcad 软件的图形编辑菜单与图形编辑

当鼠标的左键点击主菜单窗口中的 Edit 时，就会进入图形编辑状态。此时菜单窗口出现图形编辑菜单，其内容及含义如下：

EDIT	图形编辑菜单标识
Text-Edit	编辑输入文本
Copy	复制图形菜单
Move	移动图形菜单
Mirror	图形的水平镜像菜单
Rotate	图形的旋转菜单
Delete	删除图形菜单
Delete all	删除所有图形
Save Macro	子图形存盘
Load Macro	载入存盘子图形

注意该软件早期版本中可能没有 Mirror 和 Rotate 菜单选项. 本小节中仅介绍 Text-Edit 、Copy 、Save Macro 和 Load Macro 菜单选项, 其它选项的操作都是类似的, 读者可模仿使用.

§4.8.3.1 图形编辑菜单中的 Text-Edit 选项

当用鼠标的左键点击选项 Text-Edit 时，信息窗口中将出现如下的内容：

```
Pick Box/Text:
```

表示要使用者在绘图窗口中选择要编辑的文本. 在图形窗口中，所有的文本都以大写的字母 T 表示其位置，移动鼠标使得定位的十字中心靠近要编辑的 T ，然后按鼠标左键，表示要选择这个 T 所代表的文本. 这时会出现选择成功和不成功两种可能性，可分别处理如下：

- 如果选取成功，该 T 就会变成红色，同时会在信息窗口中增加如下信息：

  ```
  Edit[y/n]?
  ```

  表示要使用者确认是否要编辑该文本。按鼠标左键或键盘上的键 y 或回车键表示确认，此时屏幕的左上角将会出现如下的信息：

  ```
  Text:
  ...
  ```

其中 ... 是 T 所代表的文本，可以按照 §4.8.2.1 子节中介绍的方法修改该文本；按鼠标的右键或键盘上的键 n，表示不编辑 T 所代表的文本。

- 如果选取不成功，在信息窗口中会增加如下的信息：

  Not found, ENTER!

  表示没有成功选择到要编辑的对象，按键盘上的回车键或鼠标左键，表示结束此次选择。

§4.8.3.2 图形编辑菜单中的 Copy 菜单

当用鼠标的左键点击选项 Copy 时，菜单窗口中将出现如下的内容：

COPY	复制菜单标识
Pick objects	复制对象
Pick area	复制区域
Unpick	取消选择对象

图形的复制分两步来完成，第一步是选择要复制的子图，第二步是通过确定位移向量来复制所选择的对象。下面分别介绍这两步的具体操作过程。

1. 选择要复制的子图。有两种选择子图的方法，分别通过选项 Pick objects 和 Pick area 来实现，下面分别加以介绍。

   (a) 当我们选择 Pick objects 时，信息窗口会出现如下的信息：

   Pick object:

   表示要使用者在绘图窗口中选择要复制的对象。移动鼠标，将定位十字的中心靠近要复制的对象，然后按鼠标的左键，表示选取该对象。这时会出现选择成功和不成功两种可能性，可分别处理如下：

   - 如果选取成功，在信息窗口中增加如下信息：

     Pick[y/n]?

     表示要使用者确认此选择。按鼠标左键或键盘上的键 y 或回车键表示确认，该图形对象就会变成红色。

   - 如果选取不成功，在信息窗口中会增加如下的信息：

     Not found, ENTER!

     表示没有成功选择到要复制的对象，按键盘上的回车键或鼠标左键，表示结束此次选择。

   可以重复上述步骤选择所有需要复制的对象全体。一旦要复制的对象全体都变成了红色，可以按鼠标的右键，表示已经完成了所有要复制的对象的选择，此时信息窗口中没有任何信息。

   (b) Pick area 选项的操作和 Pick objects 选项的操作类似，只是选择复制对象的方式不同。这里是通过一个矩形框来选择复制对象，我们称这个框为选择框，当图形完全落在选择框里时表示选择该图形。当选取 Pick area 选项时，信息窗口将出现如下的信息：

   Pick area:

   Lower left corner:

表示要使用者确定选择框的左下角坐标。移动鼠标使得定位十字线的中心位于适当的位置，然后按鼠标的左键以完成选择框左下角的选择。此时信息窗口会增加如下的信息：

```
Upper right corner:
```

表示要使用者确定选择框的右上角坐标。向右上方移动鼠标，在绘图窗口会出现一个矩形框，该矩形框的右上角会随着鼠标的移动而改变，当想要复制的子图完全落在这个矩形内时，按动鼠标的左键。此时选择落在矩形内的所有子图都会变成红色，表示这些子图都成为要复制的对象。接下来可以继续选择其它复制对象。

(c) 当选取 Unpick 选项时，将会取消所有的选择对象，此时绘图窗口中的所有红色标记的图形都恢复成原来的颜色。

2. 复制所选择的子图对象。该软件是通过位移向量来控制所选对象的复制位置。在选取完复制对象后，按鼠标的右键，信息窗口会出现如下的信息：

```
Moving-vector
first point
```

表示要使用者选取位移向量的起点。移动鼠标使得定位十字线的中心位于适当的位置，然后按鼠标的左键以完成位移向量起点的选择。此时信息窗口会增加如下的信息：

```
Second point
```

表示要读者选取位移向量的结束点。移动鼠标使得定位十字线的中心位于适当的位置，然后按鼠标的左键以完成所选择对象的复制。

§4.8.3.3 图形编辑菜单中的 Save Macro 选项

当用鼠标的左键点击选项 Save Macro 时，菜单窗口中将出现如下的内容：

SAVE MACRO	子图存盘菜单标识
Pick objects	复制对象
Pick area	复制区域
Unpick	取消选择对象

要将绘图窗口中的某一些子图存盘，必须首先把它选择出来，选择的方法和 §4.8.3.2 子节中相同。在绘图窗口中，已经被选择的子图的颜色都变为红色。选择好子图后，按动鼠标的右键，信息窗口中就会出现如下的信息：

```
Filename
...\*.MAC
```

其中 ... 表示当前的路径。该信息表示让使用者输入具体的文件名，作为存储子图的文件名。用键盘修改上面的路径和文件名（注意通常不要修改扩展名 .MAC ），按鼠标左键或回车键就完成了子图的存盘。

§4.8.3.4 图形编辑菜单中的 Load Macro 选项

当用鼠标的左键点击选项 Load Macro 时，信息窗口中将出现如下的内容：

```
Filename:
```

```
...\*.MAC
```

其中 ... 表示文件的路径。通过键盘输入已存盘的图形文件（通常文件扩展名是 .MAC ），按动鼠标的左键，图形文件中所代表的图形就会以红色显示在绘图窗口的左下角，同时信息窗口会出现如下的信息：

```
Moving-Vector
First point:
```

这表示要使用者确定位移向量的起始点。移动鼠标使得定位十字线的中心位于适当的位置，然后按鼠标的左键以完成位移向量起点的选择。此时信息窗口会增加如下的信息：

```
Second point:
```

表示要读者选取位移向量的结束点。移动鼠标使得定位十字线的中心位于适当的位置，然后按鼠标的左键以将红色的图形移动到相应的位置。

注意，当信息窗口中的信息为

```
Filename:
...\*.MAC
```

时，即文件的名为 * 时，按动鼠标的左键，菜单窗口中会显示 ... 所代表的路径中所有的扩展名为 .MAC 的文件列表，可以从该列表中用鼠标直接选取所要载入的图形文件。

§4.8.4 texcad 软件的屏幕显示比例缩放菜单与图形屏幕显示缩放

当用鼠标的左键点击主菜单窗口中的 ZOOM 时，就会进入图形屏幕显示缩放状态。此时菜单窗口出现屏幕显示比例缩放菜单，其内容及含义如下：

ZOOM(...)	屏幕显示比例缩放菜单标识 ... 表示当前放大因子
Factor	改变显示放大因子
Area	屏幕局部放大
All	屏幕显示比例因子恢复到 1.00
Last	返回到前一次的缩放结果

这里只简单介绍 Factor 和 Area 选项的使用，读者可自行试验其它两个选项的效果。

§4.8.4.1 屏幕显示比例缩放菜单中的 Factor 选项

当用鼠标的左键点击选项 Factor 时，信息窗口中将出现如下的内容：

```
Zoom-factor:
...
```

其中 ... 是当前的放大因子数，可以用键盘修改这个放大因子，然后按回车键或鼠标左键表示确认。

§4.8.4.2 屏幕显示比例缩放菜单中的 Area 选项

当用鼠标的左键点击选项 Area 时，信息窗口中将出现如下的内容：

```
Zoom area:
Lower left corner:
```

表示要使用者确定被放大矩形的左下角坐标。此时移动鼠标，把定位十字的中心放到适当的位置后按动鼠标的左键，信息窗口将增加一条信息

```
Upper right corner:
```

表示要使用者确定被放大矩形的右上角坐标。此时移动鼠标，屏幕上会出现一个矩形，该矩形的右上角的坐标会随鼠标的移动而改变，当这个矩形恰好是所要放大的区域时，按动鼠标的左键，则会把矩形区域中的图形放大到整个绘图窗口。

§4.8.5 texcad 软件的绘图窗口内容平移菜单与窗口内容平移

当用右鼠标的左键点击主菜单窗口中的 Window 时，就会进入绘图窗口内容平移菜单，此时菜单窗口内容及含义如下：

WINDOW	绘图窗口内容平移菜单标识
Lower left	定位绘图窗口的左下角坐标
Upper right	定位绘图窗口的右上角坐标
Origin	绘图窗口复原

读者可以自己试用上面几个选项的效果。

§4.8.6 texcad 软件的图形存盘

在主菜单中，用鼠标的左键点击选项 Save ，信息窗口会出现如下的信息：

```
Filename:
...
```

其中 ... 为路径文件名，利用键盘输入满意的文件名（通常要把文件的扩展名取为 .pic ），然后按动回车键或是鼠标的左键，就可把绘制的图形按此文件名存盘。

§4.8.7 texcad 软件的载入图形

在主菜单中，用鼠标的左键点击选项 Load ，信息窗口会出现如下的信息：

```
Filename:
...\*.PIC
```

其中 ... 表示文件的路径。通过键盘输入已存盘的图形文件（通常文件扩展名是 .PIC ），按动鼠标的左键，图形文件所储存的图形就会显示在绘图窗口。

注意，当信息窗口中的信息为

```
Filename:
...\*.PIC
```

时，即文件的名为 * 时，按动鼠标的左键，菜单窗口中会显示 ... 所代表的路径中所有的扩展名为 .PIC 的文件列表，可以从该列表中用鼠标直接选取所要载入的文件。

§4.8.8 texcad 软件的退出

连续按动鼠标的右键，直到菜单窗口中显示的是主菜单，再用鼠标的左键点击 Quit 选项，就可以退出该程序。

§4.9 LaTeX 2ε 中的插图

尽管 LaTeX 2ε 提供了一些简单的绘图功能，但是这些绘图功能还是有一些限制，例如对圆直径、对箭头直线和直线斜率的限制等。另外，对于一些比较复杂的绘图，LaTeX 就显得无能为力了。为了解决这些问题，LaTeX 2ε 提供了专门的接口，使得人们可以在 LaTeX 2ε 中直接调用一些商业图形软件的图形格式。本节介绍在 LaTeX 2ε 中直接插入外部图像及打印输出的方法。在下面的第一小节中介绍源文件中插入图形的命令，在第二小节中介绍如何把含有插图的文件显示和打印出来的方法。

§4.9.1 在 LaTeX 2ε 源文件中插入 PostScrip 图形的方法

为了能够在 LaTeX 2ε 源文件中插入外部图形，必须调入绘图宏程序包。在众多的图形宏程序包中，graphics 和 color 是最成熟的两个包。本小节中主要介绍如何调入程序包 graphics 和 color，以及这两个程序包中的插图命令。

§4.9.1.1 在 LaTeX 2ε 源文件调用图形文件包

在源文件中调用程序包 graphics 和 color 的命令如下：

```
\usepackage[dvips]{graphics,color}
```

这里选项 dvips 表示以后将要用 dvips.exe（或 dvips32.exe）程序将 .dvi 转换成 PostScript 格式图形文件 .ps，以供 PostScript 浏览器来显示和打印排版结果(详见下一节)。在源文件里，应该在 \begin{document} 之前调入这两个图形宏程序包。调入宏程序包 graphics 和 color 之后，就可以在正文部分插入图形了，下面介绍一些插图命令。

§4.9.1.2 在 LaTeX 2ε 源文件中插入剪裁图形

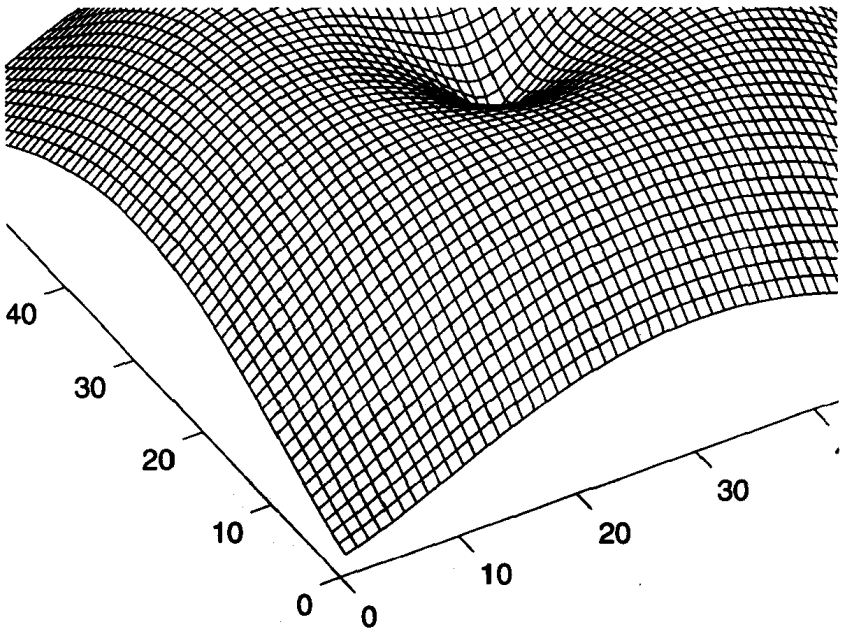
本子节介绍如何在图形文件中剪裁出一个矩形，并把该矩形插入到排版结果中的方法。可以借助于下面的命令把从图形文件剪裁出的矩形插入到排版结果中：

```
\includegraphics*[bx,by][ux,uy]{...}
```

其中 ... 是要插入的图形文件全名，bx 和 by 分别是剪裁矩形的左下角的横坐标和纵坐标，ux 和 uy 分别是剪裁矩形的右上角的横坐标和纵坐标。例如，现在有一个外部图形文件 tps.ps，它是函数 $z=\sqrt{x^2+y^2}$ 的 ps 格式图像，则

```
\begin{center}
\includegraphics*[5cm,5cm][15cm,15cm]{tps.ps}
\end{center}
```

表示调入这个图形文件的以 $(5cm, 5cm)$ 为左下角、以 $(15cm, 15cm)$ 为右上角的矩形之内部分，其打印输出的结果为：



当剪裁矩形的左下角坐标与图形的左下角坐标相同时，可以省略这个坐标的输入，例如

```
\begin{center}
\includegraphics*[10cm,10cm]{tps.ps}
\end{center}
```

表示仅调入这个图形文件的以 $(0cm, 0cm)$ 为左下角、以 $(10cm, 10cm)$ 为右上角的矩形之内部分。当图形文件的内部包含有图形尺寸的信息时，可以省略剪裁坐标的输入，此时表示调入整个图形。注意 ps 格式的图像不包含有这样的信息，因此在调用该格式的图形文件时不能省略剪裁坐标的输入；而 eps 格式的图像含有图形尺寸信息，从而在调用该格式的图形文件时可以省略剪裁坐标的输入，以表示调用整个图形。

还可以去掉剪裁出矩形区域图形命令中的 * ，即

```
\includegraphics[bx,by][ux,uy]{...}
```

其中各个符号的含义与剪裁出矩形区域图形命令完全相同，只是这一条命令把图形文件的全部都插入到排版结果中，剪裁矩形仅起到图形定位的作用。

§4.9.1.3 在 LaTeX 2ε 中把插图缩放成给定的尺寸

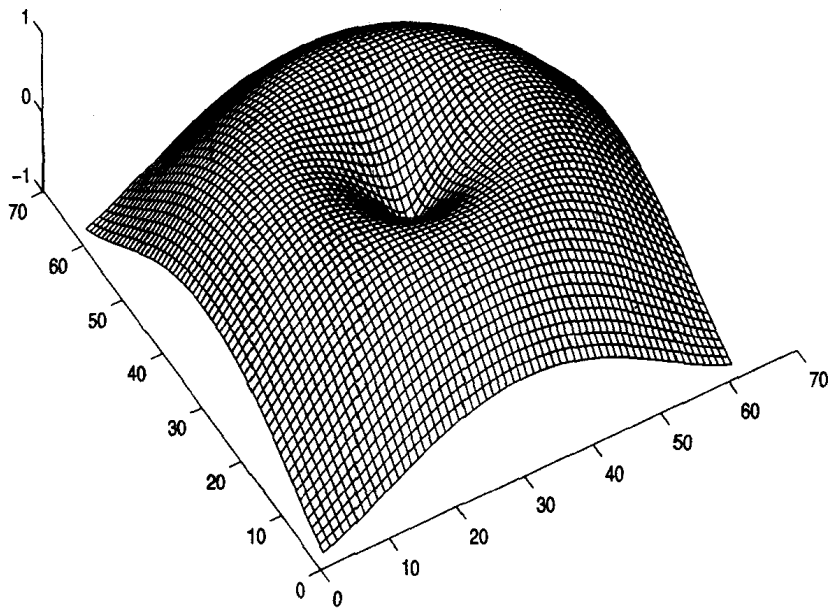
可以把调入图形文件的尺寸缩放成给定的尺寸，其方法如下：

```
\resizebox{w}{h}{...}
```

其中 w 表示缩放后图形的宽， h 表示缩放后图形的高， ... 代表插入图形的命令。例如，现在有一个外部图形文件 teps.eps ，它是函数 $z = \sqrt{x^2 + y^2}$ 的 eps 格式图形，则

```
\begin{center}
\resizebox{10cm}{10cm}{     \includegraphics{teps.eps}
}     \end{center}
```

的打印输出为



当图像的原尺寸不是矩形时，上面的命令有可能导致图像变形. 为了解决这一问题，可以只指定缩放后图形的宽，而高用 ! 代替；也可以只指定缩放后图形的高，而宽用 ! 代替. 这样就有如下的图形文件缩放命令：

```
\resizebox{w}{!}{...}
```

和

```
\resizebox{!}{h}{...}
```

其中 w 表示缩放后图形的宽， h 表示缩放后图形的高， ... 代表插入图形命令。

§4.9.1.4 在 LaTeX 2ε 中把插图按比例缩放

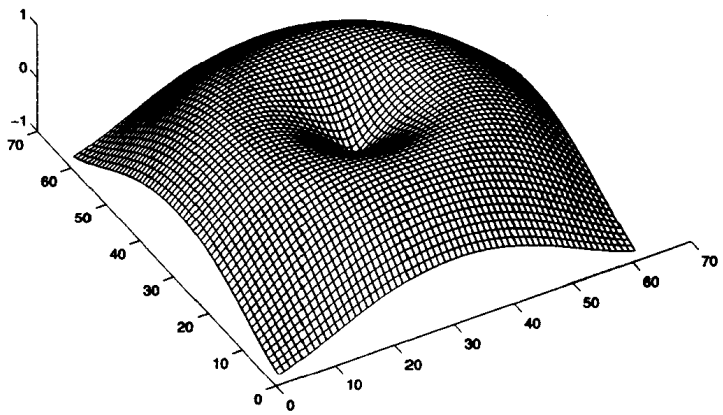
还可以按比列因子缩放所插入的图形，其命令格式如下：

```
\scalebox{w}[h]{...}
```

其中 w 为横向缩放因子， h 为纵向缩放因子， ... 代表插入图形命令. 在上面的命令中，省略纵向缩放因子相当于纵向的缩放因子等于横向的缩放因子. 例如

```
\begin{center}
\scalebox{0.5}{     \includegraphics{teps.eps}
}     \end{center}
```

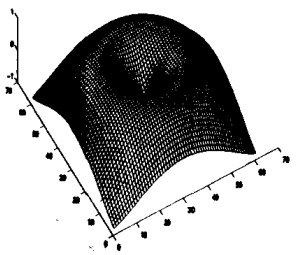
的打印输出为



相当于把 test.eps 中的图形缩小到 0.5 倍。又如

```
\begin{center}
\scalebox{0.2}[0.3]{     \includegraphics{teps.eps}
}     \end{center}
```

的打印输出为



相当于把 test.eps 中的图形的横向和纵向分别缩小到 0.2 倍和 0.3 倍.

§4.9.1.5 LaTeX 2ε 中插图的水平反射

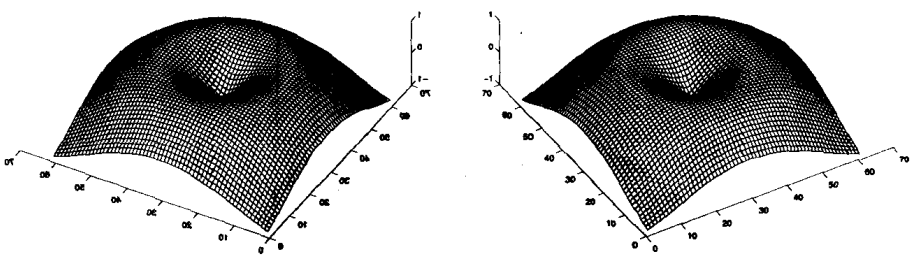
下面介绍插入图形的水平反射命令,

```
\reflectbox{...}
```

其中 ... 为插入图形的命令。例如

```
\begin{center}
\reflectbox{\scalebox{0.3}{\includegraphics{teps.eps}}}
\quad
\scalebox{0.3}{\includegraphics{teps.eps}}
\end{center}
```

的打印输出为



其中左边的图形是右边图形的水平反射。

§4.9.1.6 LaTeX 2ε 中插图的旋转

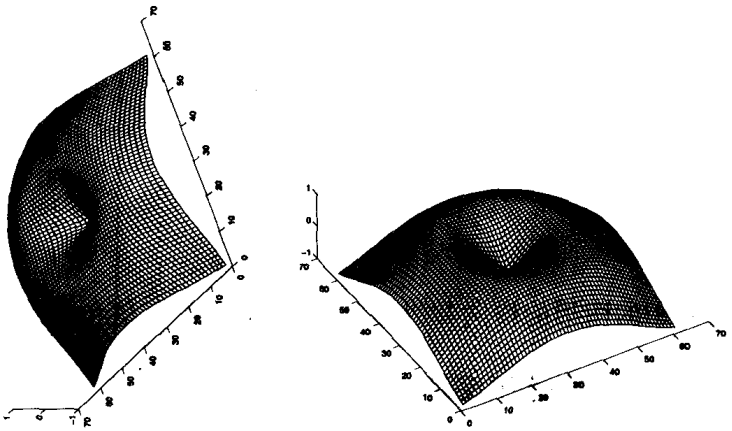
现在介绍图形的旋转命令：

```
\rotatebox{a}{...}
```

其中 a 为要旋转的角度，... 为插图命令。此命令把图形按逆时针方向旋转 a 度。例如

```
\begin{center}
\rotatebox{90}{\scalebox{0.3}{\includegraphics{teps.eps}}}
\quad
\scalebox{0.3}{\includegraphics{teps.eps}}
\end{center}
```

的打印输出为



其中左边图形是右边图形逆时针旋转 90 度的结果。

§4.9.2 把 .dvi 文件转换成 ps 格式文件的方法

当源文件中带有 §4.9.1 子节中的插图命令时，用 LaTeX 编译这个源文件也会产生 .dvi 文件，但是无论使用屏幕显示还是用打印机打印，在插图的地方只是留下了一个空白，并没有图形出现，其原因是目前的 TeX 驱动程序并不支持插入的图形文件格式。为了解决这个问题，需要把 .dvi 文件转换成 .ps，然后就可以利用支持 PostScript 的浏览器（例如 GSview 浏览器）来显示和打印所插入图形的排版效果。

最常用的把 .dvi 格式文件转换成 .ps 格式文件的工具是 dvips，它的运行方法如下：

```
dvips32 [d:][path]filename↩
```

其中 d 表示源文件所在的驱动器，path 表示源文件所在的路径，filename 为源文件的全名。当源文件的扩展名为 .dvi 时，可以省略扩展名。例如，当我们想要把当前目录下的 test.dvi 转换成 test.ps 时，可以在 Dos 的命令行中键入

```
dvips32 test↩
```

需要注意的是当 .dvi 包含汉字时，上面的转换所生成的 .ps 文件不能显示和打印汉字，其原因是转换程序找不到汉字库。为了解决这一问题，可以用 CCT 程序包附带的工具 patchdvi.exe 将带汉字的 .dvi 转换成标准的英文格式 .dvi 文件，同时生成临时汉字库，以供标准的英文 TeX 程序使用。工具 patchdvi.exe 的使用方法如下：

```
patchdvi -y -r*** file1.dvi file2.dvi↩
```

其中 *** 为解析度，file1 代表输入文件的名，file2 为输出文件的名。例如，在当前的目录下有汉字 dvi 文件 test.dvi，想把它转换成标准英文格式的 .dvi 文件 test2.dvi 时，可以在 Dos 的命令行中键入

```
patchdvi -y -r600 test.dvi test2.dvi↩
```

为了使用方便，可以编写一个批处理文件 ctxtops.bat，其内容如下：

```
if exist %1.ctx call cct %1
call latex2e %1
@echo off
call patchdvi -y -r600 %1.dvi ~~tmp.dvi
dvips32 ~~tmp.dvi
if exist %1.ps del %1.ps
ren ~~tmp.ps %1.ps
del ~~tmp.dvi
echo on
```

用这个批处理文件可以直接把 .ctx 文件编译成 .ps 文件，其使用方法如下：

```
ctxtops file↩
```

其中 file 为 .ctx 文件的名（不包括扩展名）。要注意的是 -r 后面的数字表示解析度，它一定要和 dvips 的参数文件

```
emtex\data\dvips\config.ps
```

中的解析度（即 D 后边的数字）一致。另外，当 emtex 没有生成相应解析度的英文字库时，也不能把 .dvi 文件转换成 .ps 文件。这时需要用程序 makepk.exe 来生成英文字库，具体方法如下：

```
makepk ~~~.dvi -r*** -e↩
```

这里 ~~~ 为文件名，而 *** 为要生成字库的解析度。例如

```
makepk test.dvi -r300 -e↩
```

表示以 300 解析度生成 test.dvi 文件中包含的所有字体。

§4.10 LaTeX 2ε 中的长表格

在第 129 页 §4.4.4.3 子节中我们已经介绍过几种制造表格的环境，但是 LaTeX 2ε 把这种环境中的所有内容都当作一个整体来处理，不容许在它们的内部分页，因此用这样的环境打印出长达几页的表格还比较麻烦，需要人工处理分页。在本节中介绍的编排表格的方法可以使 LaTeX 2ε 能够自动地分页，特别适合长表格的处理。还需要指出的是本节介绍的方法不仅适用于 LaTeX 2ε，还适用于 TeX 和 $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX。

打印长表格的命令结构如下：

```
\halign
 {
\hfil # \hfil && \quad \hfil # \hfil\cr
... & ... & ... \cr
... & ... & ... \cr
}
```

其中 \hfil # \hfil && \quad \hfil # \hfil\cr 称为序行，其各个符号的具体含义与第 63 页 §2.8.1 小节中序行的含义相同； ... & ... & ... \cr 为一个行，其中符号 & 表示列的分界符，控制词 \cr 表示一个行结束。

下面我们用例子显示长表格在源文件中的表示方法。下面一段的源文件

```
\halign{\hfil#\hfil\hspace{1cm}  && \hfil # \hfil\qquad \cr
    & $Y$ & $X_1$ & $X_2$ & $X_3$ & $X_4$ & $X_5$\cr
1 &100&0.11&0.98&0.5&1&0.765\cr
2 &103&0.23&1.29&9.34&2.34&0.1\cr
3 &200&22&33&55&2.3&45\cr
4 &500&2.34&3.14&2&0&99\cr
5 &20&0.9&2&10&98.5&123\cr
6 &120&0&1&23&42&1\cr
7 &320&1&4.25&6.47&5.21&192.4\cr
8 &128&3.267&4&1.097&5.432&0\cr
9 &40&0.01&0.34&0&0.2&2\cr
10 &500&0&2&79.21&1&0.45\cr
11 &100&0.11&0.98&0.5&1&0.765\cr
12 &103&0.23&1.29&9.34&2.34&0.1\cr
13 &200&22&33&55&2.3&45\cr
14 &500&2.34&3.14&2&0&99\cr
15 &20&0.9&2&10&98.5&123\cr
16 &120&0&1&23&42&1\cr
17 &320&1&4.25&6.47&5.21&192.4\cr
18 &128&3.267&4&1.097&5.432&0\cr
19 &40&0.01&0.34&0&0.2&2\cr
20 &500&0&2&79.21&1&0.45\cr
```

```
21 &100&0.11&0.98&0.5&1&0.765\cr
22 &103&0.23&1.29&9.34&2.34&0.1\cr
23 &200&22&33&55&2.3&45\cr
24 &500&2.34&3.14&2&0&99\cr
25 &20&0.9&2&10&98.5&123\cr
26 &120&0&1&23&42&1\cr
27 &320&1&4.25&6.47&5.21&192.4\cr
28 &128&3.267&4&1.097&5.432&0\cr
29 &40&0.01&0.34&0&0.2&2\cr
30 &500&0&2&79.21&1&0.45\cr
31 &100&0.11&0.98&0.5&1&0.765\cr
32 &103&0.23&1.29&9.34&2.34&0.1\cr
33 &200&22&33&55&2.3&45\cr
34 &500&2.34&3.14&2&0&99\cr
35 &20&0.9&2&10&98.5&123\cr
36 &120&0&1&23&42&1\cr
37 &320&1&4.25&6.47&5.21&192.4\cr
38 &128&3.267&4&1.097&5.432&0\cr
39 &40&0.01&0.34&0&0.2&2\cr
}
```

表示一个 40 行 8 列的数表。其中序行表示：第一列是左对齐，第二列是右对齐，其它各列是中心对齐；第一列与第二列之间的间隙增加 1cm ，第二列与第三列之间的距离为默认距离，其它各列之间的距离增加了 \qquad 距离。下面是对应上面这段源文件的的打印输出， LaTeX 2ε 恰在该表的内部分页。

	Y	X_1	X_2	X_3	X_4	X_5
1	100	0.11	0.98	0.5	1	0.765
2	103	0.23	1.29	9.34	2.34	0.1
3	200	22	33	55	2.3	45
4	500	2.34	3.14	2	0	99
5	20	0.9	2	10	98.5	123
6	120	0	1	23	42	1
7	320	1	4.25	6.47	5.21	192.4
8	128	3.267	4	1.097	5.432	0
9	40	0.01	0.34	0	0.2	2
10	500	0	2	79.21	1	0.45
11	100	0.11	0.98	0.5	1	0.765
12	103	0.23	1.29	9.34	2.34	0.1
13	200	22	33	55	2.3	45
14	500	2.34	3.14	2	0	99

15	20	0.9	2	10	98.5	123
16	120	0	1	23	42	1
17	320	1	4.25	6.47	5.21	192.4
18	128	3.267	4	1.097	5.432	0
19	40	0.01	0.34	0	0.2	2
20	500	0	2	79.21	1	0.45
21	100	0.11	0.98	0.5	1	0.765
22	103	0.23	1.29	9.34	2.34	0.1
23	200	22	33	55	2.3	45
24	500	2.34	3.14	2	0	99
25	20	0.9	2	10	98.5	123
26	120	0	1	23	42	1
27	320	1	4.25	6.47	5.21	192.4
28	128	3.267	4	1.097	5.432	0
29	40	0.01	0.34	0	0.2	2
30	500	0	2	79.21	1	0.45
31	100	0.11	0.98	0.5	1	0.765
32	103	0.23	1.29	9.34	2.34	0.1
33	200	22	33	55	2.3	45
34	500	2.34	3.14	2	0	99
35	20	0.9	2	10	98.5	123
36	120	0	1	23	42	1
37	320	1	4.25	6.47	5.21	192.4
38	128	3.267	4	1.097	5.432	0
39	40	0.01	0.34	0	0.2	2

§4.11 LaTeX 2ε 中源文件的分解

当编辑一个长 LaTeX 2ε 源文件时，可以把它分成几个独立的小文件来编写，然后通过 LaTeX 2ε 命令将它们组合在一起，本节介绍具体的实现办法。

在 55 页第 §2.7.2 节中介绍的控制词 \input，完全可以用来实现源文件的分解工作。在这里再介绍几个 LaTeX 2ε 源文件中用于同样目的的命令。首先介绍的命令是

```
\include{...}
```

其中 ... 为文件名。该命令只能用在正文中，并且当文件的扩展名为 .tex 时，可以省略这个扩展名。这个命令表示把一个外部文件作为源文件正文的一部分，并且总是把外部文件的内容安排到新的一页上开始打印。

在 LaTeX 2ε 中，有一个和 \include 配套使用的命令，它可以决定是否让 LaTeX 2ε 在编译源文件时是否重新编译由 \include 所引入的外部文件。该命令的具体结构如下：

```
\includeonly{...,...,...}
```

其中用逗号隔开的各个 ... 表示各个不同的外部文件名,如果这些外部文件的扩展名为 .tex ，可以省略扩展名。该命令告诉 TeX 仅编译在其参量中列出的那些由 \include 命令导入的外部文件。另外，该命令只能用在源文件中的版面定义部分里，即只能用在 \begin{document} 之前。

下面看一个例子。假设有一个名为 test.tex 的源文件，其内容如下：

```
\documentclass{book}
...
\includeonly{ch1,ch5}
...
\begin{document}
\include{ch1}
\include{ch2}
\include{ch3}
\include{ch4}
\include{ch5}
\include{ch6}
\end{document}
```

这说明在源文件 test.tex 中引入了外部文件： ch1.tex、ch2.tex、ch3.tex、ch4.tex、ch5.tex 和 ch6.tex; 其中 \includeonly{ch1,ch5} 告诉 LaTeX 2ε 在编译 test.tex 时仅编译其中的 ch1.tex 和 ch5.tex。如果我们把 test.tex 的内容改为

```
\documentclass{book}
...
%\includeonly{ch1,ch5}
...
\begin{document}
\include{ch1}
\include{ch2}
\include{ch3}
\include{ch4}
\include{ch5}
\include{ch6}
\end{document}
```

则 LaTeX 2ε 在编译 test.tex 时会重新编译所有由 \include 引入的 6 个外部文件。

之所以引入 \includeonly 和 \include 命令，是为了节省长文件的编译时间。虽然在编译源文件时能够为我们节省宝贵的时间，但这样做的代价是：只能使得 \includeonly 后列出的文件的内容得到更新。要特别注意，一旦我们已经定稿，一定要在 \includeonly 命令的前面加上一个百分号 % ，使得该命令失效，然后再重新连续两次编译源文件，以得到最后的修改结果。在使用上， \input 和 \include 命令的区别主要有如下两点：

1. 由 \include 命令引入的源文件中不能再包含它本身,即 \include 命令不可以嵌套使用。

但是 TeX 命令 \input 却可以嵌套使用。

2. 由 \include 命令引入的源文件的内容必须在新的一页开始排版。由 \input 命令引入的源文件的内容没有这种限制。

因此人们在书的排版中，通常建立一个主源文件，其内容和前面的例子差不多，然后把各章作为一个独立的源文件来编写，通过 \inclue 命令把各章源文件引入到主源文件中。本书就是按照这种方法建立源文件的。

§4.12 LaTeX 2ε 中索引的建立

当我们编写一本书时，如果想手工为全书的一些名词做一个索引，将十分困难。因为全书排版定稿后才能确定各个名词所在的页码，在此基础上才能做索引工作。如果在做好索引后修改书中的部分内容，需要再次详细核对整个索引中名词页码是否正确。正是因为用人工做索引的诸多不便，以至于很多的书籍都不包含一个全书的索引。

LaTeX 2ε 提供了建立索引的命令，并能自动生成一个包含所有标注过的名词和名词所在页码的文件(该文件与源文件同名，扩展名为 .idx ，为称呼简单，我们称这个文件为 .idx 文件)，通过手工将 .idx 文件中的名词分类及排序就可以建立索引了。由此可见 LaTeX 2ε 自身所提供的编写索引的方法还是很麻烦的，特别是每次修改正文内容将导致以前的有关索引的排工作失效。为了方便为 LaTeX 2ε 建立索引，Pehong Chen 在 Leslie Lamport 的帮助下编写了 makeindx.exe 程序，它可以把 .idx 文件直接转换成 LaTeX 2ε 的索引格式源文件，从而避免了人工将名词分类和排序等繁杂的工作。本节简单介绍利用 makeindx.exe 编写索引的方法。

§4.12.1 LaTeX 2ε 源文件中的索引纪录

要让 LaTeX 2ε 建立 .idx 文件，先要在源文件中加入索引纪录，这是通过如下的带有参量的控制词来完成的：

```
\index{...}
```

其中 ... 为索引条目。索引条目有如下的结构：

```
m!sm!ssm
```

其中 m 为主条目， sm 为子条目， ssm 为子子条目。为了叙述简单，我们把主条目、子条目和子子条目简称为条目。条目是由文本组成，可以包括各种符号和 TeX 命令。另外， ! 、 @ 、 | 和 "(除了 \" 中的 " 之外，因为 \" 总是表示德语中的重音，详见第 25 页中的 §2.4.3 小节)在索引条目中有特殊的含义，因此在主条目、子条目和子子条目中分别用 "!、"@、"| 和 "" 表示这些符号。下面简单介绍条目文本的构成语法：

1. 为了能够控制条目的排序方式， LaTeX 2ε 提供了如下的条目文本结构：

   ```
   ...@***
   ```

   其中 ... 和 *** 都是文本，并且 @ 的前面不是 "。@ 之前的文本只用作排序，并不在索引中出现，我们称它为排序条目文本。 @ 之后的文本是索引条目中实际显示的文本，我们称它为显示条目文本。在一个条目文本中，至多只能出现一个前面不带 " 的 @。如果条目文本中没有出现前面不带 " 的 @，则该条目文本既是排序条目文本，又是显示条目文本。

2. 在索引中，想要条目文本的后面给出的页码是一个范围，需要在条目文本以 |(或 |) 结束。例如在源文件的两个不同的地方分别出现

```
\index{...|(} 和 \index{...|)}
```

这里两个 \index 命令中的文本 ... 是完全相同的，此时对应于这两条命令生成一个索引条目，其后面的页码按如下的格式打印：

```
n-m
```

其中 n 是第一条 \index 所在的页码， m 是第二条 \index 所在的页码。注意， |(或 |) 的后面应该是 }。

3. 条目中表示花括号的命令必须同时成对出现。例如在源文件中不能出现 \index{\{} ，却可以有 \index{\{...\}}。

4. 如果想把索引中某一个条目的后面页码替换成 see ... ，这里 ... 是另一条索引，可以用 |see{...} 作为 \index 的参量的结尾，即在源文件中采用如下的方式输入：

```
\index{***|see{...}}
```

其中 *** 和 ... 都是文本。

下面的表列出了一个源文件中的所有索引纪录和相应（利用 makeindex 生成的）索引打印输出：

第 4 页：	\index{ant}	ant, 4 family, 9 home, 6
第 3 页：	\index{big}	\begin, 8
第 6 页：	\index{ant|home}	big, 3
第 6 页：	\index{dog|home|(}	dog home, 6-7
第 7 页：	\index{dog|home|)}	emmet, *see* ant
第 8 页：	\index{begin@\verb=\begin=}	
第 9 页：	\index{ant|see{ant}}	
第 9 页：	\index{ant!family}	

其中最左边列给出的是索引纪录所在的页码，竖线左边列给出了索引纪录在源文件中的表示，竖线右边给出了相应索引项的打印输出。

§4.12.2 LaTeX 2ε 源文件中 .idx 文件与索引

当在源文件中建立起索引纪录后， LaTeX 2ε 并不能在编译时生成 .idx 文件。为了能够使 LaTeX 2ε 自动生成 .idx 文件，需要在源文件的版面定义部分输入命令：

```
\makeindex
```

该命令通知 LaTeX 2ε 在编译源文件时生成 .idx 文件。

.idx 文件只是按照索引纪录在源文件中的次序给出了对应于各个索引纪录的显示文本和

对应的页码的纪录，需要源文件的编辑者去对这些记录进行排序、编辑等大量的工作才能完成索引部分的源文件的建立。在 LaTeX 2ε 源文件中，索引用如下的索引环境表示：

```
\begin{theindex}
\item ...
\subitem ...
\subitem ...
\subsubitem ...
\item ...
\item ...
\end{theindex}
```

其中各个 ... 代表各个索引显示文本。这里 \item、\subitem 和 \subsubitem 后的 文本分别表示主条目、子条目和子子条目索引文本。另外，

```
\indexspace
```

在索引环境中表示空白的一行。下面的索引打印输出



```
ant, 4
    family, 9
    home, 6

\begin, 8

big, 3
dog
    home, 6-7
```

在源文件中的表示为：

```
\begin{theindex}
\item ant, 4
\subitem family, 9
\subitem home, 6
\indexspace
\item \verb=\begin=, 8
\indexspace
\item big, 3
\item dog
\subitem home, 6-7
\end{theindex}
```

虽然 LaTeX 2ε 提供了建立索引的方法，但是通过这种方法建立索引的工作量是很大的，并且正文的修改将可能使前次建立的索引的页码标注错误。为克服上述缺陷，可以使用下一小节中的方法自动让 LaTeX 2ε 建立索引。

§4.12.3 makeindex 工具与 LaTeX 2ε 的自动索引

为方便索引的建立， Pehong Chen 在 Leslie Lamport 的帮助下编写了 makeindx.exe 程序，它可以把 .idx 文件直接转换成 LaTeX 2ε 的索引环境格式源文件 .ind 。其实 .ind 文件的

内容就是一个包括了所有索引纪录项的 theindex 环境，将该文件内容复制到 LaTeX 2ε 源文件的相应地方，就可以表示索引了。 makeindx.exe 软件的使用方法如下：

```
makeindx filename↩
```

其中 filename 为 .idx 文件名。

上面的方法需要在 LaTeX 2ε 源文件中复制 .ind 文件的内容，只是半自动地建立索引。为了能够自动建立索引，可利用宏程序包 makeidx.sty 中提供的命令，具体的实施步骤如下：

1. 在 LaTeX 2ε 源文件的版面定义部分增加下面一行

```
\usepackage{makeidx}
```

2. 在 LaTeX 2ε 源文件中需要插入索引的地方添加如下的命令：

```
\printindex
```

实际上，在 makeindex 宏程序包里，定义了命令 \printindex ，他表示读入外部 .ind.

为了能够完全自动生索引，可以编写一个批处理文件 latexidx.bat ， 其内容如下：

```
if exist %1.ctx call cct %1
@echo off
tex &latex %1 %2 %3 %4 %5
makeindx %1
tex &latex %1 %2 %3 %4 %5
```

在上面的批处理文件中，对源文件编译了两次，其目的是为了得到最后对源文件的修改结果。在 Dos 环境下使用该批处理文件格式如下

```
latexidx filename↩
```

其中 filename 为源文件名.

第五章　中文 TEX

TEX 是英文软件，为了能够用它来处理中文，很多人作了不少工作，开发出一些软件。其中最为流行的是由中国科学院计算数学与科学工程计算研究所张林波等人开发的 CCT 系统和华东师范大学等高校的数学工作者开发的“天元”（ Ty ）系统，使得人们可以在 TEX 中处理中文，并且现在这两个软件还是免费提供的。读者可以在如下的网址

ftp://ftp.cc.ac.cn/pub/cct/msdos/

下载 CCT 系统软件；可以在华东师范大学的网址

ftp://wims.math.ecnu.edu.cn/pub/ty/tywin???.zip

下载 Windows 版本的天元系统软件，其中“ ??? ”是一些数字，标志天元系统的版本号（目前的最新版本是 2.21 版）。

本书是用 CCT 编写的。在下面的 §5.1 和 §5.2 中，我们简单介绍 CCT 的使用方法，有关 CCT 的更详细使用方法，见该软件附带的《 CCT DOS 版参考手册》；在 §5.3 中简单介绍在 Windows9x 中有效使用 TEX 的方法；最后在 §5.4 中简单介绍天元系统。

§5.1　CCT 系统里中文 TEX 源文件的建立与编译流程

为了叙述方便，称用 CCT 规则编写的中文 TEX 源文件为 CCT 源文件。 CCT 源文件和 TEX 源文件、 $\mathcal{AMS}$-TEX 源文件以及 LaTeX 2ε 源文件的构造基本相同，只是在源文件的开头要调入一个宏程序包，以使 TEX 能兼容中文。在 TEX 和 $\mathcal{AMS}$-TEX 源文件中调入该宏程序包的命令为：

```
\input cchead.sty
```

在 LaTeX 2ε 源文件中，调入该宏程序包的命令为：

```
\usepackage{cchead}
```

对于 LaTeX 2ε 源文件来说，上面的这个宏程序包虽然可以解决汉字的兼容问题，但是汉字在像标题、页眉等环境中的大小却还需要人为控制，一个更好的方法是直接调用中文类文件。编排中文科技图书时可以调用中文书类 cctbook ，此时源文件的类选项行应该是：

```
\documentclass[...]{cctbook}
```

其中 ... 代表类选项。编排中文科技论文可以选用中文论文类 cctart ，此时源文件的类选项行应该是：

```
\documentclass[...]{cctart}
```

其中 ... 代表类选项。上面两个中文类的类选项与英文类的类选项完全相同，详见 106 页中 §4.2.1 中的介绍。通常 CCT 源文件的扩展名为 .ctx ，有时也把这种 CCT 源文件称为 .ctx 文件。

§5.1.1　CCT 中的汉字字体与字号

在宏程序包 cchead.sty 中定义了一些与汉字有关的 TEX 命令，下面列出几条有关汉字字体和字号的命令及含义。

\zihao{n}	选择汉字字号，其中 n 为一个代表字号的整数，可以取的值为： 0 、 1 、 2 、 3 、 4 、 -4 、 5 、 -5 、 6 和 7 。其中 0 代表初号汉字； 1 、 2 、 3 、 4 、 5 、 6 、 7 分别代表一、二、三、四、五、六、七号汉字； -4 和 -5 分别代表小四和小五号汉字。
\songti	选择宋体字。
\heiti	选择黑体字。
\kaishu	选择楷书字体。
\fangsong	选择仿宋字体。
\biaosong	选择标宋字体。
\ziti{...}	选择字体的命令，其中 ... 为大写字母 A 、 B 、 C 、 D 、 E 等。这里 A 、 B 、 C 、 D 、 E 分别代表宋体、黑体、楷书、仿宋和标宋，其它字母的含义由 ccfonts.def 中定义。

§5.1.2 CCT 源文件的编译

CCT 源文件要先转换为 TEX 源文件，然后根据源文件所采用的格式选用 TEX 、 AMS-TEX 或 LaTeX 2ε 进行编译即可。下分几段具体介绍。

§5.1.2.1 plain 、 vanilla 和 ltr 格式 .ctx 文件的编译

plain 、 vanilla 和 ltr 格式 CCT 源文件可按如下步骤编译：

1. cct filename ↩
2. tex filemane ↩

其中 filename 是源文件的名（其扩展名 .ctx 不用输入）。经过上面两步就可以生成 .dvi 文件，然后就可以用 TEX 的标准屏幕显示驱动程序或打印驱动程序进行屏幕显示或打印输出。

§5.1.2.2 AMS-TEX 格式 .ctx 文件的编译

AMS-TEX 格式 CCT 源文件可按如下步骤编译：

1. cct filename ↩
2. amstex fiemane ↩

其中 filename 是源文件的名（其扩展名 .ctx 不用输入）。经过上面两步就可以生成 .dvi 文件，然后就可以用 TEX 的标准屏幕显示驱动程序或打印驱动程序进行屏幕显示或打印输出。

可以用通过修改批处理文件 amstex.bat ，使得它既可以编译英文 AMS-TEX 源文件，同时也可以编译中文 CCT 源文件。修改后的批处理文件 amstex.bat 的内容如下：

```
if exist %1.ctx call cct %1
@echo off
tex &amstex %1 %2 %3 %4 %5
```

把修改后的批处理文件 amstex.bat 拷贝到 emtex 软件所在的目录下，就可以用 82 页 §3.1 节中的方法编译 CCT 源文件。

§5.1.2.3 LaTeX 2ε 格式 .ctx 文件的编译

plain 、 vanilla 和 ltr 格式 CCT 源文件可按如下步骤编译：

1. cct filename ↩
2. latex fiemane ↩

其中 filename 是源文件的名（其扩展名 .ctx 不用输入）. 经过上面两步就可以生成 .dvi 文件，然后就可以用 TEX 的标准屏幕显示驱动程序或打印驱动程序进行屏幕显示或打印输出.

可以用通过修改批处理文件 latex.bat ，使得它既可以编译英文 LaTeX 源文件，同时也可以编译中文 CCT 源文件. 修改后的批处理文件 latex.bat 的内容如下：

```
if exist %1.ctx call cct %1
@echo off
tex &latex %1 %2 %3 %4 %5
```

把修改后的批处理文件 latex.bat 拷贝到 emtex 软件所在的目录下，就可以用 106 页 §4.1.2 小节中的方法编译 CCT 源文件.

§5.2 在 window9x 环境中显示 .dvi 文件

为在微软公司的 windows 环境下显示和打印 .dvi 文件，张林波又开发了 cctwin16.exe 和 cctwin32.exe 程序. 在 win32 系统中， cctwin32.exe 的速度高于 cctwin16.exe 。为了使用这个程序，需要将它复制到 emtex 系统的目录下.

§5.3 在 window9x 环境中编写和编译 TEX 源文件

在本节中我们以 emtex(3.14159[4a] 版）为例介绍 Dos 版本的 TEX 在 Windows9x 环境中的使用方法. 因为 emtex 是 Dos 环境的 TEX 软件，所以只能在 windows9x 中的 Dos 环境来编译源文件，并在 Dos 环境来显示和打印编译出的 .dvi 文件. 通过这种途径在 windows9x 环境中使用 Dos 版本的 emtex 软件是比较麻烦的，一个解决的途径是使用 Windows 版本的 TEX 软件包，如 PCTeX32 或 Scientific WorkPlace 软件包，但问题是还不知道如何在这些软件里使用中文.

有一些在 Windows9x 中的编辑软件可以运行 Dos 程序，如 EditPlus. 可以利用这种功能把 emtex 、 CCT 和 cctwin32 组合成一个集成的 TEX 工作环境，使得人们能够非常方便地在这个集成环境中编写和编译 TEX 源文件，显示和打印 .dvi 文件. 编辑软件 EditPlus 是一个共享软件，在下面的网址上可以下载该软件的 30 天试用版本：

http://www.editplus.com

下面分几步简单介绍如何建立和使用这种集成环境。

1. 在 Windows 启动时设置 emtex 工作环境。
   (a) 在“开始”菜单上选择“运行 (R)... ”菜单；
   (b) 在“打开 (O) ”窗口中输入“ sysedit ”后按确认按钮；

(c) 此时系统会运行“系统配置编辑程序”,该程序同时打开如下的系统配置文件: protocol.ini 、 system.ini 、 win.ini 、 config.sys 和 autoexec.bat;

(d) 察看一下 autoexec.bat 文件的内容,看它是否包括了如下的一行:

```
d:\emtex\setenv
```

这里 d 是你的 emtex 软件所在的盘符。如果没有, 请添加这一行后存盘并退出“系统配置编辑程序”; 如果已经存在这一行, 可退出“系统配置编辑程序”(此步骤是针对 emtex 系统 3.14159[4a] 版的, 其它的版本可能不需要此步骤)。

2. 在 emtex 目录中建立批处理文件 “ clatex.bat ”, 其内容如下:

```
@echo off
if exist %1.ctx call cct %1
tex &latex %1 %2 %3 %4 %5
```

3. 安装 editplus 软件

4. 设置 editplus 软件中的 LaTeX 2ε 编译、屏幕显示及文件类型工具。运行 editplus 软件, 用鼠标选择顶行菜单的 “ tools ” 选项, 在下拉菜单中选取 “ Preferences... ” 选项。这时 window 系统会弹出设置菜单 “ Preferences ”, 在此菜单中按下面的方法设置工具。

(a) 设置 LaTeX 2ε 编译工具:

i. 在弹出菜单 “ Preferences ” 的左边的 Categories 框中, 选择 User tools 行;

ii. 点击右边的 “ Add ” 按钮, 在弹出的小菜单中选 “ Program ”;

iii. 在右下方的 “ Menu text ” 框中加入文本 : Compile LaTeX

iv. 按 “ Command ” 框右边的 “…” 钮;

v. 在弹出的对话窗口中的“搜寻”窗口中选取 emTeX 文件夹, 在文件窗口中选择批处理文件 Clatex.bat(注意如文件窗口中列出的文件太多, 可改变文件类型框中的内容, 以便于搜索 *.bat 文件), 点击“打开”按钮;

vi. 在弹出菜单 “ Preferences ” 中点击 “ Argument ” 框右边的按钮, 在弹出的菜单中选择“ File Name Without Extension ”项, 这时在 “ Argument ” 框中自动插入文本 : $(FileNameNoExt);

vii. 在右下角的四个复选窗中, 点击 “ Close window on exit ” 前的复选窗, 使该复选窗中出现对号, 其它复选窗为空白。

(b) 添加 .dvi 文件察看工具:

i. 再按弹出菜单 “ Preferences ” 的 “ Add ” 按钮;

ii. 在右下方的 “ Menu text ” 框中加入文本: View Dvi

iii. 按 “ Command ” 框右边的 “…” 钮;

iv. 在弹出的对话窗口中的“搜寻”窗口中选取 emTeX 文件夹，在文件窗口中选择文件 cctwin32.exe(注意如文件窗口中列出的文件太多，可改变文件类型框中的内容，以便于搜索 *.exe 文件)，点击“打开”按钮；

v. 在弹出菜单“ Preferences ”中点击“ Argument ”框右边的按钮，在弹出的菜单中选择“ File Name Without Extension ”项，这时在“ Argument ”框中自动插入文本 ： $(FileNameNoExt)；

vi. 在右下角的四个复选窗中，点击“ Close window on exit ”前的复选窗，使该复选窗中出现对号，其它复选窗为空白。

(c) 添加文件类型：

i. 在弹出菜单“ Preferences ”的左边的“ Categories ”框中，选取“ Filters ”行；

ii. 点击右边的“ Add ”按钮；

iii. 在 Description 框中加入文本 ： emTeX

iv. 在“ Wild cards ”框中加入文本 ： *.ctx;*.tex

v. 在右下角的几个复选窗口中，点击“ Add to Common Files filter ”前的复选窗，使该复选窗中出现对号，其它复选窗为空白。

(d) 设置编辑器的显示字体（如果你觉得该编辑器的字体的大小不合适才需要调整此项）：

i. 在弹出菜单“ Preferences ”的左边“ Categories ”框中，选取“ Fonts ”行；

ii. 在“ Area ”选择窗口中选择你想要调整的字体所在区域（例如“ Default (Edit Window) ”）；

iii. 在“ Font ”窗口中选取需要的字体；

iv. 在“ Font style ”窗口中选取需要的字体格式；

v. 在“ Size ”窗口中选取需要的字号。

(e) 点击弹出菜单“ Preferences ”下面的“ OK ”按钮。

现在我们已经在 editplus 中建立好了编写、编译中英文 LaTeX 2ε 源文件的集成环境，并且还集成了显示 .dvi 文件的环境。下面介绍一下如何使用这个集成环境。可以用 editplus 建立 LaTeX 源文件，然后用如下的几种方法编译这个源文件：

- 用快捷键方式编译源文件：同时按下 ctrl 键和数字键 1（或者是某一个其它的数字键，可以在 Tools 的下拉菜单中发现具体的数字键是什么）。
- 同时按 Alt 键和字母 t 键，这时会出现一个下拉菜单，在该菜单的下面部分，你会发现“ Compile LaTeX ”选项，该选项的右边标出了这个选项的快捷键（例如当该选项右边标出的是“ ctrl+1 ”就表示“ Compile LaTeX ”选项的的快捷键是同时按下 ctrl 键和数字键 1 ），用鼠标点击该选项， editplus 就会调用 LaTeX 2ε 编译编辑窗口的源文件。

- 用鼠标点击顶行菜单上的 " Tools ", 这时会出现一个下拉菜单, 在该菜单的下面部分, 你会发现 " Compile LaTeX " 选项, 用鼠标点击该选项, editplus 就会调用 LaTeX2ε 来编译编辑窗口的源文件.

如果我们已经用上面的三种方式之一编译了编辑窗口中的源文件, 就可以按如下提供的几种方式利用 cctwin32.exe 在屏幕上显示编译出的 .dvi 文件:

- 用快捷键方式显示源文件: 同时按下 ctrl 键和数字键 2 (或者是某一个其它的数字键, 可以在 Tools 的下拉菜单中发现具体的数字键是什么).
- 同时按 Alt 键和字母 t 键, 这时会出现一个下拉菜单, 在该菜单的下面部分, 你会发现 " View Dvi " 选项, 该选项的右边标出了这个选项的快捷键 (例如当该选项右边标出的是 " ctrl+2 " 就表示 " View Dvi " 选项的的快捷键是同时按下 ctrl 键和数字键 2), 用鼠标点击该选项, editplus 就会调用 cctwin32.exe 显示编译好的 .dvi 文件.
- 用鼠标点击顶行菜单上的 " Tools ", 这时会出现一个下拉菜单, 在该菜单的下面部分, 你会发现 " View Dvi " 选项, 用鼠标点击在选项, editplus 就会调用 cctwin32.exe 显示编译好的 .dvi 文件.

注意, 当我们察看完 LaTeX2ε 的屏幕显示效果后, 不必马上关闭 cctwin23.exe 程序, 你可以切换到 editplus 程序继续编辑或修改源文件, 当你再次想看修改的结果时, 可以按前面介绍的方法再次地编译源文件, 然后切换到刚才没有关闭的 cctwin32.exe 程序窗口, 这时窗口的内容会自动更新. 因此我们可以按照需要在 editplus 程序和 cctwin32.exe 程序之间任意切换, 修改源文件和察看修改效果, 这为使用者提供了极大的方便. 另外, 我们还可以利用 " View Dvi " 选项同时打开多个 cctwin32.exe 程序窗口, 以显示 .dvi 文件的不同页的内容.

§5.4 天元系统简介

天元排版软件是由华东师范大学等高校的数学工作者编写的配合各种版本 TeX 使用的中文科技排版软件, 由于其早期版本的编写得到天元数学基金的部分资助而取此名. 它是使用 TeX 前的中文预处理软件, 主要功能是辨认源文件的中文字符, 把它们翻译成西文 TeX 可以处理的编码, 同时把用到的中文编成 TeX 格式的字符库, 供屏幕显示和打印使用. 由于该软件是在使用 TeX 前外加的预处理, 它可以与任何格式和版本的 TeX 配合使用, 也可以用任何输出设备输出排版的结果 (只要 TeX 提供相应的驱动程序) .

目前天元排版软件有 Windows 版与 DOS 版两个版本. 它们的用法和功能大致相同, 同一源文件的排版结果完全相同, 因此这里只介绍 Windows 版 TYWIN 的用法.

§5.4.1 TYWIN 的特点

TYWIN 可以处理用扩展国际汉字码即 GBK 码 (GB13000) 写成的源文件. GBK 码就是扩展的国标码, 它包含 21000 多个汉字, 原来的国标码 (GB2312) 只包含 6000 多个汉字, 是 GBK 的一个子集. GBK 码中包括了所有的繁体汉字、异体字、古体字以及日文和韩文中的汉字. 但

是为了把国标码以外的汉字打印出来，必须利用中文 Windows9x 的宋体和黑体字库或其他含大字集的字库。用户可以自定义汉字大小。

肖刚开发的 TeXDraw 已被嵌入 TYWIN ，能够画出参数曲线以及各种精确的数学图像。

TYWIN 取消了对输出密度的限制，例如若把密度设为 1016DPI ，依次用天元和 TEX 处理后，把生成的 DVI 文件用 DVIPS 输出成 PS 文件，就可以在照排机上出片，达到专业级的印刷质量。

§5.4.2 TYWIN 的安装与卸载

TYWIN 软件可以用匿名 ftp 从华东师范大学数学系 WIMS 网站

wims.math.ecnu.edu.cn

的 pub/ty 目录下找到。这里天元系统是 zip 形式的压缩文件，文件名是 tywin???.zip ，其中“ ??? ”是一些数字，标志 TYWIN 的版本号。目前的最新版本是 2.21 版。安装时先建立一个子目录（如 tywin），然后把压缩包里的文件释放到该目录中去。将该目录下所有文件和子目录（连同其文件）都拷贝到 TEX 软件所在的目录中。现在 TEX 软件所在目录中的 user 子目录下面的文件是试验用的例子。可以在桌面建立一个指向 tywin.exe 的快捷方式。

卸载时只要把这些前面拷贝到 TEX 目录及子目录中的所有文件删去即可，不会留下垃圾。压缩包内的 readme.txt 是一个中文文本文件，包含了安装信息， tywin.hlp 则包含了如何使用天元的信息。

§5.4.3 TYWIN 的配置

点击 TYWIN 快捷方式进入 TYWIN 的窗口后，再点击“ Config ”按钮，就进入配置对话框。配置对话框有 5 页：“路径”、“字模文件名”、“字模与结束方式”、“源文件”以及“汉字大小”。大部分选项的意义是自明的。这里仅对以下几项加以说明。

1. 更新 Tab 文件：选中后即重新生成 TYCFNT.TAB 文件，从而所有的 TFM 文件以及 PK 文件均被更新。当 TYWIN 运行时间长了，累积的汉字太多时就有必要加以更新。
2. PK 文件路径：请填入 C:\TeXFONTS\DPI@Rr\ 或 C:\TeXFONTS\@RrDPI\ 的形式， TYWIN 会自动把数字填入。如果所需的 DPI??? 子目录不存在，则会自动建立。如果您需要用 PXL???? 形式的子目录，则请填入 PXL@RVr 。
3. 字模文件名（仅对使用 TTF 字模有效）：在 16 种字体命令的右边各有一个小框及大框。小框是用来修改字体命令的。例如你若想把“ \姚”改名为“ \篆”，只需在小框内输入“篆”字即可。大框用来输入相应的 TTF 字模的名称及路径。 TTF 字模的完整文件名 = 第一页“字模文件路径” + 大框填写的路径及文件名。例如若在“字模文件路径”中填入了 c:\windows\fonts\ ，又在相应的大框中填入了 simsun.ttf ，那么这个字模文件的完整名称是 c:\windows\fonts\simsun.ttf 这样你可以使用分布于不同子目录里的字模文件。没有使用的字体可以不设定字

模文件名。如果你对于不同的文件使用不同的字体命令以及字模文件，则建议你把配置文件与天元源文件一起保存，以保证将来编译时能重现当前的设置。

4. 汉字大小：本页供有特殊需要的用户使用。可以设定汉字的字宽、字深、字间距以及瘦体系数和阔体系数。汉字总是方的，字宽减去字深就是字高. 框内数字置 0 意为取默认值。每次修改后所有的汉字会重新生成。默认值对应于原 TYWIN 的设定。

§5.4.4 天元源文件的编写

如果您使用的是 PlainTeX 或 AMSTeX ，那么在出现第一个汉字之前要加上命令：

```
\input tyinput
```

文件 TYINPUT.tex 是天元软件自动生成的，与输出的 TEX 文件位于同一个子目录. 如果您使用的是 LATEX 或 LATEX 2ε ，那么请在文件的开头输入以下两行：

```
\def\ChineseScale{????}
\input tyinput
```

然后才是 \documentclass 或 \documentstyle 。其中的 ???? 是一个整数，必须等于您在配置对话框中为 \magnification 设定的值。以后就可以像英文一样输入中文.

天元的命令仅用于描述汉字字体、字形和大小，都是在“ \ ”符号以后紧跟一个汉字. 天元命令可以分为以下三类：

1. 确定汉字尺寸的命令有：

\半(=\五)=5pt	\六=6pt	\七=7pt
\八(=\小)=8pt	\九=9pt	\标=10pt（默认）
\中=12pt	\大=14pt	\特=17pt
\双=20pt	\巨=25pt	\叁=30pt
\肆=40pt	\伍=50pt	\陆=60pt

2. 天元的字体命令有以下几种，在使用 UCDOS3.0 的矢量字模时分别对应于下列字体：

\宋：宋体（默认）	\楷：楷体	\仿：仿宋体
\黑：黑体	\题：标题宋	\行：行楷
\粗：粗黑体	\姚：姚体	\美：美黑
\圆：细圆	\准：准圆	\魏：魏碑
\变：隶变	\隶：隶书	\琥：琥珀
\综：综艺		

如果使用 TTF 字模，则字体命令仅与您提供的对应字模文件有关，与命令的字面含义无关。您也可以修改上列字体命令，以与真实字型相符。

3．字形命令： \瘦、 \阔、 \扁：分别使字形变瘦、变阔和变扁。 \正：恢复正常字形

与 TEX 的命令一样，要使天元的命令只在局部起作用，应把它放在一对花括号 {...} 中；也可用同类型的命令去改变前一个命令的作用。注意， TEX 的其他命令对于天元的命令都是无效的。 TEX 中所有定义字体、字形或字体大小的命令对于天元都是无效的，因此，要使版面中的中外文字体、字型及字的大小相称，必须对中外文字体、字型和字的大小分别给指令。

在压缩包的 \TeXuser 子目录中给出了几个天元源文件的例子，供用户参考．在源文件准备好后，打开 TYWIN ，如果不需改动配置，则只要点击“ Open ”按钮，选中要处理的天元源文件，就会自动生成同名的 TEX 文件．也可以用行命令方式

```
tywin filename
```

调用天元，处理完毕后会自动关闭窗口。

§5.4.5 在 TEX 中用 TeXDraw 绘图

TeXDraw 是肖刚设计的在 TEX 环境里使用的绘图软件，其特点是能利用 TEX 画出精确的插图。为了使用 TeXDraw ，必须先把 TeXDraw 的 font 使用的 *.tfm 文件以及 *.pk 文件复制入相应子目录．把 TDW.TeX 以及 TDWC.TeX 复制入子目录 TeXinput 中．这些文件均已包含在压缩包内．

如果在源程序中使用了 TeXDraw ，则必须在前面加入语句

```
\input tdwc
```

(当使用 AMSTeX 或 PlainTeX 且 \magnification=1095 时) ，或

```
\input tdw
```

(其它情形) 。具体的画图命令读者参看压缩包内提供的相关文档。

第六章　$\mathcal{AMS}$-TEX 和 LATEX 快速入门

在本章中介绍 $\mathcal{AMS}$-TEX 和 LATEX 的几个模板，使得初学者能够在短时间内掌握这些格式的中英文源文件的结构。对于不愿意了解细节的读者来说，本章能够帮助它们迅速掌握编写源文件的一般规则。对于有 TEX 使用经验的读者来说这些模板也是有益处的，可以在这些模板的基础上构造自己的模板，这样在编写新文件时只需填入标题及新内容即可。

以反斜杠 \ 开始的字符串都是 TEX 、$\mathcal{AMS}$-TEX 和 LATEX 命令，例如 \TeX 就是一个命令，它代表 TEX 。由 TEX 、 $\mathcal{AMS}$-TEX 和 LATEX 命令和文本内容组成的文件分别称为 TEX源文件、$\mathcal{AMS}$-TEX 源文件和 LATEX 源文件。在所有的这些源文件中， $...$ 结构表示正文中的数学公式， $$...$$ 结构表示单列数学公式。例如 $x_{ij}y^{2}=\alpha_{ij}$ 表示 $x_{ij}y^2=\alpha_{ij}$ ，而 $$x_{ij}y^{2}=\alpha_{ij}$$ 表示

$$x_{ij}y^2=\alpha_{ij}$$

在数学公式中， ^{...} 表示上标， _{...} 表示下标， \alpha 表示希腊字母 α 。在源文件中，所有的小写希腊字母都用 \ 和紧跟其后该字母的英文单词表示。在源文件中， % 表示同行位于该百分号后面的内容为注解。

§6.1　$\mathcal{AMS}$-TEX 快速入门

在本节中，分别给出 $\mathcal{AMS}$-TEX 源文件的 amsppt 格式中英文的模板，该模板对于国外数学杂志投稿者来说是有用的。在该模板中，百分号 % 后面的中文是中文注解。读者可以阅读这些模板中的中文注解，以理解所涉及的 TEX 命令的含义。进一步，读者还可以自己动手建立该模板，并用第 82 页 §3.1 节中介绍的方法编译该模板。

§6.1.1　英文 amsppt 格式模板

下面给出的是英文 amsppt 格式模板，其文件的扩展名为 tex。

```
\magnification=\magstep1% 文章的整体放大比列
% 其选项可以是 \magstephalf,\magstep1,\magstep2,...,\magstep5
\documentstyle{amsppt}% 选用的格式
\pagewidth{15cm}% 版面的宽
\pageheight{42pc}% 版面的高
\NoRunningHeads% 不打印页眉
\pageno=3% 设置第一页的页码为 3

% 在此处输入其它的控制版面的命令

% 在此处输入自定义命令
```

```
\topmatter
\title
The Title is Book% 此行是文章的标题
\endtitle

\author
author1\\ % 第一作者，其中的双反斜杠表示换行
author2 % 第二作者
\endauthor

\affil
(Beijing Normal University) % 单位
\endaffil

\address
address1 % 通信地址
\endaddress

\email
name\@bnu.edu.cn % 电子邮件地址
\endemail

\dedicatory
Dedicated to % 奉献
\enddedicatory

\date
December, 10, 1999% 日期
\enddate

\thanks
Thanks% 致谢
\endthanks

\translator
lll. A % 译者名
\endtranslator
```

```
\keywords
ads, ghj% 关键词
\endkeywords

\subjclass
60J25% 分类号
\endsubjclass

\abstract
Big man...% 摘要
\endabstract
\endtopmatter

\document

\head  \S1. This is head  % 节标题
\endhead

% 正文内容

\proclaim{Theorem 1.1}
This is a paper.% 定理内容
\endproclaim

\demo{Proof}
% 证明内容
\enddemo

% 正文内容

\Refs
% 输入参考文献

\ref
\no
1% 参考文献编号
\by
author% 作者名
```

```
\paper
Title% 文章标题
\jour
Journal% 杂志名称
\yr
2000% 年
\pages
20-100% 页码
\endref

\widestnumber\key{GHMaR}% 括号中是参考文献中最长的关键词

\ref\key
GHMaR% 参考文件的关键词
\by author% 作者名
\book
Book% 书名
\bookinfo
Press% 出版社等
\yr
2000% 年
\endref

\endRefs

\enddocument
```

§6.1.2 中文 amsppt 格式模板

下面给出的是中文 amsppt 格式模板，其文件的扩展名为 .ctx 。由于在诸如表示文章的标题和页眉等地方的命令里，中文的兼容性并不太好，因此在下面的模版中采取了变通的方式在这些命令里输入中文. 另外，像 \abstract 和 \Ref 这类的命令会产生一个英文输出，因此不适于用在中文的 .ctx 源文件中。

```
\magnification=\magstep1% 文章的整体放大比列
% 其选项可以是 \magstephalf,\magstep1,\magstep2,...,\magstep5
\documentstyle{amsppt}% 选用的格式
\input cchead.sty% 使源文件能够兼容中文的宏程序包
\pagewidth{15cm}% 版面的宽
\pageheight{42pc}% 版面的高
```

```
\NoRunningHeads% 不打印页眉
\pageno=4% 设置第一页的页码为 4

\font\mytitlefont=cmr10 scaled\magstep2

\def\myTitle{\heiti\zihao{4} \mytitlefont We say
文章的标题 }% 此行是文章的标题
\def\myLeftHead{\zihao{-5}
左页眉标题 }% 此行是左页眉标题
\def\myRightHead{\zihao{-5}
右页眉标题 }% 此行是右页眉标题
\leftheadtext{\myLeftHead}
\rightheadtext{\myRightHead}

% 在此处输入其它的控制版面的命令

% 在此处输入自定义命令

\topmatter
\title
\myTitle
\endtitle

\author
作者 % 作者，其中的双反斜杠表示换行
\endauthor

\affil
北京师范大学数学系，北京， 100875 % 单位
\endaffil

\address
北京师范大学数学系 % 通信地址
\endaddress

\dedicatory
献给我的 ... % 献词
\enddedicatory
```

```
\date
1999 年 10 月 10 日 % 日期
\enddate

\thanks
感谢 % 致谢
\endthanks

\endtopmatter
\document
{\narrower
{\heiti  摘要 } \zihao{-5}
这里是输入摘要的地方...    % 这里是输入摘要的地方
\par}

\head  \S1.  第一节 \endhead% 节标题
这里是输入第一节正文的地方.  % 正文内容

\subhead \S1.1  第一小节 \endsubhead
这里是输入第一小节正文的地方.  % 正文内容

\subsubhead \S1.1.1  第一小段 \endsubsubhead
这里是输入第一小段正文的地方.  % 正文内容

\midspace{1.5cm}
\centerline{\mytitlefont\heiti  参考文献 }

\item{[1]}  第一条参考文献

\item{[2]}  第二条参考文献

\enddocument
```

§6.2 LaTeX 2ε 快速入门

在本节中，分别给出 LaTeX 2ε 源文件的中英文的模板。在这些模板中，百分号 % 后面的中文是中文注解。读者可以阅读这些模板中的中文注解，以理解所涉及的 LaTeX 2ε 命令的含义。进一步，读者还可以自己动手建立这些模板，并用第 106 页 §4.1.2 节中介绍的方法编译这些模板（要先用第 178 页 §5.1.2.3 小节的方法修改批处理文件 latex.bat ，否则不能编译 CCT 源文件）。

§6.2.1 英文 LaTeX 2ε 模板

下面给出的是英文 LaTeX 2ε 模板，此时文件的扩展名为 .tex 。

```
\documentclass[11pt]{article}% 文件类说明
% 还可以选择的类是 book 或 report

\setlength{\parindent}{12pt}% 自然段第一行的缩进量为 12pt
\setlength{\parskip}{10pt plus1pt minus2pt}
% 自然段之间的距离为 10pt ，并可在 8pt 到 11pt 之间变化
\setlength{\baselineskip}{8pt plus2pt minus1pt}
% 行间距为 8pt ，并可在 7pt 到 10pt 之间变化
\setlength{\textheight}{21true cm}% 版面高为 21 厘米
\setlength{\textwidth}{14.5true cm}% 版面宽为 14.5 厘米

\begin{document}% 正文开始

\title{Thesis}% 文章标题，双反斜杠 \\ 表示换行
\author{author\\ Dept. of Math.}
% 作者名，单位，通信地址等，双反斜杠 \\ 表示换行
\date{1999/11/04}
% 文章写作日期，如果省略此行，计算机日期作为写作日期

\maketitle% 建立标题部分

% 文章的正文输入

\end{document}% 源文件的结束
```

§6.2.2 中文 LaTeX 2ε 模板

下面给出的是中文 LaTeX 2ε 模板，此时文件的扩展名为 .ctx 。

```
\documentclass[11pt]{cctart}% 文件类说明
% 还可以选择的类是 cctbook

\setlength{\parindent}{12pt}% 自然段第一行的缩进量为 12pt
\setlength{\parskip}{10pt plus1pt minus2pt}
% 自然段之间的距离为 10pt , 并可在 8pt 到 11pt 之间变化
\setlength{\baselineskip}{8pt plus2pt minus1pt}
% 行间距为 8pt , 并可在 7pt 到 10pt 之间变化
\setlength{\textheight}{21true cm}% 版面高为 21 厘米
\setlength{\textwidth}{14.5true cm}% 版面宽为 14.5 厘米

\begin{document}% 正文开始

\title{Thesis}% 文章标题, 双反斜杠 \\ 表示换行
\author{author\\ Dept. of Math.}
% 作者名, 单位, 通信地址等, 双反斜杠 \\ 表示换行
\date{1999/11/04}
% 文章写作日期, 如果省略此行, 计算机日期作为写作日期

\maketitle% 建立标题部分

% 文章的正文输入

\end{document}% 源文件的结束
```

第七章　TEX 软件包的相关网站及最新发展简介

Donald Knuth 开发 TEX 原始程序的主要目的是打印英文数学文章。随着时间的推移，TEX 的优点迅速吸引了人们的注意，它在全世界科技界和学术界的影响越来越广。1991 年 Knuth 停止了对 TEX 程序在功能上的进一步开发，转而把精力集中在完善该程序的稳定性。但是他同意其它人以 TEX 原码为基础进一步扩展它的功能，这导致了各具特色的、不同名称的 TEX 版本。

本章先介绍一些与 TEX 有关的网站，以方便读者从 Internet 上了解 TEX 的最新发展动向；然后简单介绍运行于 Windows 环境中的 TEX 软件包 MikTeX 的一些有趣的特色；最后简单介绍 Windows 环境中的 TEX 源文件的编辑软件 WinEdt。

§7.1　与 TEX 有关的一些网站

在美国、英国和德国有 TEX 的三个主要的网络服务器，这些服务器成为 TEX 程序的主要来源。这三个服务器不但各自随时更新其内容，而且还相互定期交换信息，读者可以在这些网站上找到最新版本的 TEX 软件、字库、宏程序包等。这三个网络服务器的地址如下：

国家	网址	IP 地址
美国	ftp://pip.shsu.edu	192.92.115.10
英国	ftp://ftp.tex.ac.uk	134.151.79.32
德国	ftp://ftp.dante.de	129.206.100.192

这三个网站的目录结构基本相同。在 index 文件中对当前目录的每一个子目录的目的进行了简单的介绍。笔者曾经利用微软公司的 IE5.5 成功地登陆到德国的网站。

在这些 TEX 专用网络服务器上，一个最有用的目录是：

/tex-archive/

它的下面的各个子目录中存放了各种与 TEX 有关的软件。例如：

/tex-archive/systems/win32/

下存放的是运行于 Windows 环境的各种版本 TEX 软件，其中

/tex-archive/systems/win32/miktex/

存放的则是在下一节将要介绍的 MikTeX 软件包。

§7.2　运行于 Windows 环境的 TEX 软件包：MikTeX

MikTeX 是运行于 Windows 环境中的免费 TEX 版本，它包括了 TEX 程序包以及相关的一些应用程序。该软件由 Christian Schenk 维护开发，读者可以到该软件的主页

http://www.miktex.org/

了解有关该软件的最新信息。

MikTeX 包含了许多有用的 TEX 软件包，并且提供了详细的文档，读者在成功安装该软件后，可以在该软件的系统目录下找到子目录 DOC 。在该子目录下，按各个程序包分类将各自的

使用简介放在相应的子目录中。本节中不想依次详细介绍 MikTeX 中各个软件包的使用，对这方面感兴趣的读者可以参考 DOC 目录下相应子目录中的介绍文档. 这里只是简单介绍如何下载及安装该软件，以及该软件所提供的一些有趣的功能。

§7.2.1 MikTeX 软件包的下载与安装

可以到如下的网址之一下载 MikTeX 软件的最新版本：

ftp://ftp.dante.de/tex-archive/systems/win32/miktex/

ftp://ftp.tex.ac.uk/tex-archive/systems/win32/miktex/

ftp://ctan.tug.org/tex-archive/systems/win32/miktex/

到 2000 年 9 月 19 日为止， MikTeX 软件的最新版本是 1.2e 。为了下载方便，我们通常下载 Zip 格式的版本，例如 miktex120.zip 就是 1.2 版压缩文档，它的长度为 22617k 字节。

将下载的压缩文档 miktex120.zip 释放(用解压软件 WINZIP)到硬盘的一个临时目录中，就可以在这个临时目录下发现一个名为 setupwiz.exe 的文件（或者在该临时目录的一个子目录下发现该文件），这个文件就是 MikTeX 的安装程序。运行该程序，按程序的提示可以完成 MikTeX 软件的安装。

§7.2.2 MikTeX 的几个有趣的特点

MikTeX 提供了专门编译 LaTeX 文件的程序 Latex.exe 和浏览 DVI 文件的程序 Yap.exe 。这两个程序提供了一些很有用的功能。其中最有趣的是它们相互配合，可以轻易地从 .dvi 文件的浏览的状态返回到源文件的编辑状态。这种功能的实现是通过 Latex.exe 在编译 LaTeX 源文件的同时产生一些有关源文件中文字的定位信息，而 Yap.exe 则可以利用这些定位信息通过双击鼠标的左键，直接从浏览状态返回到源文件编辑器中，并且会把光标自动停放在与浏览器双击点相关联的那一行源文件上. 此功能对于我们修改文章中的错误是极其有用的。下面假设 MikTeX 已经成功地安装在你的计算机上。

§7.2.2.1 在编译源文件的同时产生定位信息

使得 Latex.exe 能够产生定位信息的行命令格式如下：

```
Latex -scr filename
```

这里 filename 代表要编辑的 LaTeX 源文件名称（不包括扩展名）.

§7.2.2.2 建立源文件编辑器与 MikTeX 之间的联系

任何一个纯文本编辑软件，只要它支持执行行命令的功能，就能够与 MikTeX 关联起来，建立起集成源文件的编辑和编译以及进行屏幕显示排版结果等功能的环境。下面以编辑软件 EditPlus 为例介绍如何建立起源文件编辑器与 MikTeX 的编译程序 Latex.exe 以及 DVI 浏览器 Yap.exe 之间的这种联系的方法。

首先介绍如何使得 EditPlus 与编译软件 Latex.exe 相关联. 按照 §5.3 中介绍的方法进入 EditPlus 的 “ References ” 弹出菜单，然后按照如下的步骤添加编译 LaTeX 源文件的工具 MikLaTeX:

1. 在弹出菜单 " Preferences " 的左边的 Categories 框中，选择 User tools 行；
2. 点击右边的 " Add " 按钮，在弹出的小菜单中选 " Program "；
3. 在右下方的 " Menu text " 框中加入文本： MikLaTeX
4. 按 " Command " 框右边的 "…" 钮；
5. 在弹出的对话窗口中的 "搜寻" 窗口中选取 MikTeX 软件包所在的子目录，再进入下一级子目录 miktex ，最后进入下一级子目录 bin ，在该子目录中选中文件 Latex.exe ，点击 "打开" 按钮；
6. 此时在 " Command " 框中出现的文本应该是以 " latex.exe " 结尾的，在它的后面输入一个空格后，添加如下的字符串： -scr
7. 在弹出菜单" Preferences "中点击" Argument "框右边的按钮，在弹出的菜单中选择 " File Name Without Extension " 项，这时在 " Argument " 框中自动插入文本： $(FileNameNoExt)
8. 在右下角的四个复选窗中，点击 " Close window on exit " 前的复选窗，使该复选窗中出现对号，其它复选窗为空白。

这样就完成了 EditPlus 与编译软件 Latex.exe 相关联的操作。当在 EditPlus 中编辑完一个 LaTeX 源文件后，就可以通过顶行下拉菜单 " Tools " 中的 " MikLaTeX " 选项来编译这个源文件。

现在介绍如何使得 EditPlus 与浏览 .dvi 文件的软件 Yap.exe 相关联。按照前面类似的方法进入 EditPlus 的 " References " 弹出菜单，具体的操作过程如下：

1. 再按弹出菜单 " Preferences " 的 " Add " 按钮
2. 在右下方的 " Menu text " 框中加入文本： MikView
3. 按 " Command " 框右边的 "…" 钮；
4. 在弹出的对话窗口中的 "搜寻" 窗口中选取 MikTeX 软件包所在的子目录，再进入下一级子目录 miktex ，最后进入下一级子目录 bin ，在该子目录中选中文件 Yap.exe ，点击 "打开" 按钮；
5. 在弹出菜单" Preferences "中点击" Argument "框右边的按钮，在弹出的菜单中选择 " File Name Without Extension " 项，这时在 " Argument " 框中自动插入文本： $(FileNameNoExt)
6. 在右下角的四个复选窗中，点击 " Close window on exit " 前的复选窗，使该复选窗中出现对号，其它复选窗为空白。

现在已经完成了 EditPlus 与 Yap.exe 的关联工作。在完成对 EditPlus 中的源文件的编译工作后，就可以通过顶行下拉菜单 " Tools " 中的 " MikTeX " 选项来浏览排版结果。

§7.2.2.3 建立 Yap.exe 与源文件编辑器间的联系

如果源文件编辑器的行命令格式支持光标的定位，则可以建立起 Yap.exe 和源文件编辑器之间联系，方便使用者修改排版的结果。编辑软件 EditPlus 的行命令格式支持光标的定位，我们以它为例介绍如何建立 .dvi 文件浏览器 Yap.exe 与 LaTeX 源文件编辑器间的联系。

这需要对 Yap.exe 程序的 " Inverse Search " 标签的参数进行设置，指明 LaTeX 源文件编辑器 EditPlus 在硬盘中所在的位置及有关鼠标定位的行命令选项。这些选项设置好后，就能实现在 Yap.exe 中通过双击鼠标左键返回到源文件编辑器 EditPlus 相应的行上的功能。对 Yap.exe 程序参数进行设置的具体步骤如下：

1. 运行 Yap.exe;
2. 在 Yap.exe 的顶行菜单上找到 " View " 菜单，用鼠标的左键点击该菜单;
3. 此时会出现下拉菜单，其中有 " Options... " 选项，把鼠标移动到该选项上，然后单击鼠标左键;
4. 此时会出现一个弹出菜单，其上半部分有一些标签，其中的一个标签就是

   Inverse Search

   标签（目前 2.0beda4 版本中该标签位于最左端）。将鼠标置于该标签上后单击鼠标左键;
5. 此时在标签的下方会出现两个上下排列的窗口，上边是一个选项窗口，下边的是可以输入文本的编辑窗口。在下边的编辑窗口中按如下的方式输入参数：

   "path\editplus.exe" "%f" "-cursor" "%l:1"

   其中 " path " 代表编辑软件 editplus.exe 所在的路径。例如，当 editplus.exe 所在的路径是

   C:\Program Files\EditPlus 2

   时，参数的输入为：

   "C:\Program Files\EditPlus 2\editplus.exe" "%f" "-cursor" "%l:1"

通过上面的五个步骤，就完成了建立 .dvi 文件浏览器 Yap.exe 与 LaTeX 源文件编辑器间联系的工作。

§7.2.2.4 Yap.exe 与源文件编辑器之间的切换

按 §7.2.2.3 中的步骤设置好参数后，就可以从 .dvi 文件浏览器 Yap 直接进入源文件编辑器 EditPlus ，并且可以在 EditPlus 和 Yap 之间方便地切换，具体操作如下：

- 由 Yap 进入 EditPluse:
  1. 在 Yap 顶行菜单的下面有一个工具条，其上面有各种按钮。将手形图标形状的按钮用鼠标的左键按下;
  2. 此时屏幕光标也变成了手的形状。当按下鼠标的左键时，小手的五指会向浏览的文字抓去，此时移动光标（鼠标的左键保持按下状态），会使整个屏幕所显示的文字随小手一起平移。把鼠标放在某一行文字上后，双击鼠标的左键，Yap 就会自动进入到源文件编辑器 EditPlus 中，并自动把光标定位在双击鼠标时小手所在位置对应的那一行文本上。
- 由 EditPluse 进入 Yap:
  1. 用鼠标点击 EditPlus 顶行菜单 " Tools " 选项。在出现的下拉菜单中点击 " MikView " 选项，此时 EditPlus 就会运行 Yap（假设已经按照前面的部分成功设置了 EditPlus 的参数）;

2. 如果此时 Yap 已经运行，则可以按上一条的方法运行 Yap ，结果出现一个新的 Yap 窗口，此时浏览的是 .dvi 文件的第一页；
3. 如果此时 Yap 已经运行，也可以利用 Windows9x 的底行工具条，用鼠标直接点击表示 Yap 的按钮。这种方法的优点是不重新运行 Yap ，节省计算机的内存开销及启动 Yap 软件的时间开销，并且浏览页码的定位能够保持原来的状态。

§7.2.3 TEX 软件包 Omega 的特点

在 MikTeX 中包含了一个名为 Omega 的 TEX 软件包，该软件包是 John Plaice 开发的一个 TEX 扩展版本，它主要是扩展了 TEX 的多语言适应能力。它把原始 TEX 中用 8 位二进制数表示的字符和指针都改用 16 位二进制数表示，消除了许多 TEX 有关编译和语言使用方面的限制。 Omega 还容许多语言的输入和输出，利用可编程的控制实现从一种字符集到另一种字符集的转换。 Omega 的内部编码是通用的基于 ISO-10646 标准的 Unicode 字符集，这使得 TEX 的使用者更容易处理多重的或复杂的语言，例如阿拉伯文、印度文、高棉文、中文、日文或朝鲜文，而且还有利于在此基础上进行功能的进一步开发，诸如民族化支持和超文本特点等。在 Omega 中用 Lambda 来编译 LATEX 源文件. 标准的 Omega 版本包括了新标准编码 “ TEX Unicode ”。读者可以在 MikTeX 软件包所安装的目录下的

```
doc\omega\base
```

子目录中找到有关 Omega 软件包的详细使用方法。

§7.3 Windows 环境中的 TEX 源文件编辑器： WinEdt

我们已经知道，用 TEX 排版大体可以分为三个阶段。第一步是建立源文件，第二步是编译源文件，第三步则是模拟显示或打印输出。这其中建立和修改源文件的过程，是关键，是我们经常需要面对的过程。在 Windows9x 系统中，可以使用专门为该系统编写的文本编辑软件来编写 TEX 源程序。有很多这样的编辑软件可以选择，例如 EditPlus.exe ， Editpad.exe ， Note.exe ， Wirte.exe ， Word 等。但是这些文本文件编辑器都不是专门为 TEX 而设计的，用来编写 TEX 源文件的效率不是很高。

为了提高源文件的编写效率人们编写了许多 TEX 专用编辑器，其中有些是免费使用的，而另一些则是以共享软件的形式提供。例如 “ Wintex 2000 ”， “ WinEdt ” 等，读者可以在网址

```
http://home.germany.net/100-122054/texwin.htm
```

上找到类似的这样一些免费或共享的 TEX 源文件编辑器的列表。通常这些编辑器需要某种版本的 TEX 来编译源文件从而生成 .dvi 文件。在这些源文件编辑中， WinEdt 是一个具有较多功能的比较好用的源文件编辑器。到目前为止（ 2000 年 9 月 10 日），WinEdt 的最新版本为第 5 版，本节简单介绍它的一些特点。

§7.3.1 WinEdt 的安装

在安装 Winedt 之前，应该先安装某种版本的 TEX 软件。理论上 Winedt 支持任何版本的 TEX 软件，但是参数的设置需要随 TEX 软件版本的不同而不同。对于使用者来说，最简单的是

用 MikTeX 来配合 WinEdt 使用，因为 Winedt 能够自动识别 MikTeX 的目录结构，从而能够方便地确定它的运行参数。下面假设计算机中已经安装了 MikTeX，我们来介绍如何安装和设置 WinEdt。

可以在如下的网址之一下载 WinEdt：

```
http://www.winedt.com/
ftp://nt-newton.fho-emden.de/pub/software/pc/win/winedt32/
```

下载的文件名为 WinEdt5.exe（这是第 5 版的名称，共有 2632k 字节），它是一个自动解压的压缩文件。

为了安装该软件，需要先解压缩。运行 WinEdt5.exe，会出现一个弹出菜单，上面有一个编辑窗口，让使用者添加解压文件存放的位置。假设我们想要把它存放在 C 盘的 \temp 子目录中，可以在这个窗口中输入：

```
C:\temp
```

然后按回车键。现在 WinEdt5.exe 就会释放它所包含的文件到 C 盘的 \temp 子目录中，在该子目录中可以找到安装程序 Setup.exe，运行它就可以按照提示完成软件的安装工作。

完成 WinEdt 的安装后，通常在桌面上会出现执行该程序的图标，用鼠标的左键双击该图标，就会进入该程序。第一次运行该程序，它会自动的打开一个名为 WinEdt.txt 的文件，其内容是该软件的一个介绍。现在需要设置一下该软件的运行参数，才能使它编译 TEX 源文件和显示排版结果。参数设置可按如下步骤进行：

1. 用鼠标的左键单击顶行菜单上的“ Options ”选项，此时会出现一个下拉菜单；
2. 在下拉菜单中，把鼠标放在“ Configrations ”选项上，此时会弹出一个子菜单；
3. 在这个子菜单中，把鼠标放在“ TeX ”选项上，此时又会弹出一个菜单；
4. 在新弹出的菜单上用鼠标的左键点击“ MikTeX Dictionary ”选项。

现在就可以编译 TEX 源文件和模拟显示排版结果了。

§7.3.2 WinEdt 的功能简介

WinEdt 具有许多实用的功能，这里只简单列举几个，其详细的功能介绍请读者从该软件顶行菜单中的“ Help ”下拉菜单中查找。

1. 运行该程序，它会自动打开上次关闭它时所编辑的文件。
2. 提供了“ log ”窗口。编译 TEX 源文件时，所有的编译信息都会出现在该窗口，用鼠标的左键单击该窗口上的右（左）红箭头，该窗口的内容可以自动的跳转到下（上）一条有错误或警告信息的地方，同时编辑窗口的光标也跳转到源文件中的相应地方，这使得改错特别方便。
3. 提供了实时拼写查错功能，避免英文单词的拼写错误。
4. 可以方便地将 .dvi 文件转换成 .ps 文件。

参考文献

[1] Arvind Borde. *Mathematical TEX by example*. Boston: Academic Press, 1993

[2] Donald E Knuth. *The TEXbook*. Reading, Massachusetts: Addision-Wesley, 1984

[3] Donald E Knuth. *The METAFONTbook*. Reading, Massachusetts: Addision-Wesley, 1986

[4] David Salomon. *The advanced TeXbook*. New York: Springer,1995

[5] Eitan Gurari. *TEX and LATEX*. New York: McGraw-Hill, 1994

[6] 郭力，张林波等. CCT 中外科技激光照排系统。北京：海洋出版社， 1993

[7] Michael D Spivak. *The Joy of TEX*. Rholde Island: American Mathematical Society, 1990

[8] Michael Vulis. *Mordern TEX and its applications*. Boca Raton: CRC Press, 1993

[9] Leslie Lamport. *LATEX: A Document Preparation System*. Reading, Massachusetts: Addison-Wesley, 1994

[10] 王翼勋. 科技英语轻印刷系统应用指南。北京：学苑出版社， 1993

[11] 张林波. CCT DOS 版参考手册。 CCT 软件包， 1997

索 引

+...\cr 固定表格环境中的行结构 (vanilla), 67

((, 72
)), 72
@" 空隙调整 (vanilla), 23
@ 修正空隙 (vanilla), 23
[[, 72
!` ¡, 26
! 索引中命令符 (LaTeX 2ε), 172
" 索引中命令符 (LaTeX 2ε), 172
--- 可以分行的破折号, 24
-- 可以分行的破折号, 24
- 可以分行的破折号, 24
?` ¿, 26
@--- 禁止分行的破折号, 24
@-- 禁止分行的破折号, 24
@- 禁止分行的破折号, 24
@ 索引中命令符 (LaTeX 2ε), 172
@ TeX 保留键, 19
# 制表环境中序行里的列位置, 63
% 注释符, 5
&& 制表环境中序行里的重复处理符, 63
& 制表环境中序行里的列间隔位置, 63
{ {, 72
} }, 72
~ 禁止断行的空隙, 24
' ′, 31
`` ‘‘, 4
` ‘, 4
'' ’’, 4
' ’, 4
␣ 空格, 8
/ /, 72
\! 减少数学公式里两字母间的空隙, 49
\, 1/2 英文字符宽的空隙, 19
\. 英文缩写词标记, 23
\; 小空隙, 49
\@ @, 19
\B 重音符, 26
\D 重音符, 26
\H 重音符, 26
\# #, 8
\$ $, 8
\% %, 8
\& &, 8
_ _, 8
\{ 左花括号, 19
\} 右花括号, 19
\~ 重音符, 26
\' 重音符, 26
\` 重音符, 26
\" 重音符, 26
\\ 断行符, 19
\^ 重音符, 26
\b 重音符, 26
\c 重音符, 26
\d 重音符, 26
\sb 公式下标标志, 30
\sp 公式上标标志, 30
\t 重音符, 26
\u 重音符, 26
\v 重音符, 26
\| ‖, 71
^ 公式上标标志, 30
_ 公式下标标志, 30
|(索引条目开始页码标记 (LaTeX 2ε), 173
|) 索引条目结束页码标记 (LaTeX 2ε), 173

| 索引中命令符 (LaTeX 2ε), 172
| |, 72
$$...$$ 单列数学公式环境, 122
$$ 单列公式的开头和结尾标志, 29
$...$ 正文数学公式环境, 27, 122
*{n}{...} 表格环境中表示相同 n 个列格式的结构 (LaTeX 2ε), 131
- 数学公式中减号, 27
/ 数学公式中除号, 27
@{...} 表格环境中输入列分隔文本的结构 (LaTeX 2ε), 130
\(...\) 正文数学公式环境, 122
\- 断词符, 13
\/ 小空隙, 6
\ 命令开始符, 5
\[...\] 单列数学公式环境, 122
| 表格环境中的竖线 (LaTeX 2ε), 130
| 数学公式中竖线, 27
|| 表格环境中的双竖线 (LaTeX 2ε), 130
{} 子体, 6
]], 72

a4paper 类的纸型选项 (LaTeX 2ε), 107
a5paper 类的纸型选项 (LaTeX 2ε), 107
\AA Å, 26
\aa å, 26
abstract 摘要环境 (LaTeX 2ε), 116
\abstract...\endabstract 输入文章摘要的结构 (amsppt), 85
\acute 数学公式中的重音标记, 48
\addcontentsline{toc}{...}{...} 手动向目录表添加内容的结构 (LaTeX 2ε), 118
\address 地址, 52
\address...\endaddress 输入作者通信地址的结构 (amsppt), 85
\addtolength{...}{...} 改变长度单位所代表长度的结构 (LaTeX 2ε), 113
\AE Æ, 26
\ae æ, 26
\affil...\endaffil 输入作者单位的结构 (amsppt), 85
\aleph ℵ, 70
\align...\\...\endalign 多行数学公式 (vanilla), 40
\aligned...\\...\endaligned 局部多行数学公式 (vanilla), 41
\allowlinebreak 可分行点 (vanilla), 24
\allowmathbreak 数学公式中可以分行的点 (vanilla), 29
\Alph 大写英文字母编号 (LaTeX 2ε), 139
\alph 小写英文字母编号 (LaTeX 2ε), 139
\alpha α, 27, 69
\amalg ∐, 70
\and &, 70
\and 使双作者并列排放 (LaTeX 2ε), 115
\angle ∠, 71
\annotations 信的附言开始标志 (ltr), 52
\appendix 附录开始标志 (LaTeX 2ε), 119
appendix 附录环境 (LaTeX 2ε), 119
\approx ≈, 70
\approxeq ≊(AMS-TeX), 102
\arabic 阿拉伯数字编号 (LaTeX 2ε), 139
\arccos arccos, 38
\arcsin arcsin, 38
\arctan arctan, 38
\arg arg, 38
array 数学公式中多行对齐环境 (LaTeX 2ε), 126
article 编写论文的类 (LaTeX 2ε), 107

\ast $*$, 70
\ast 代替键盘上的 * 键, 69
\asymp $\asymp$, 70
\atop 数学公式中使字符上下排列 (LaTeX 2ε), 134
\author...\endauthor 作者输入结构 (vanilla), 19, 85
\author{...} 文章作者输入结构 (LaTeX 2ε), 115

b5paper 类的纸型选项 (LaTeX 2ε), 107
\backepsilon $\ni$ (AMS-TeX), 102
\backprime $\backprime$ (AMS-TeX), 101
\backsim $\backsim$ (AMS-TeX), 102
\backsimeq $\backsimeq$ (AMS-TeX), 102
\backslash $\backslash$, 71, 72
\bar 数学公式中的重音标记, 48
\barwedge $\barwedge$ (AMS-TeX), 101
\baselineskip 行间距 (LaTeX 2ε), 111
\baselineskip... 调整正文的行距 (vanilla), 60
\baselinestretch 决定行距的因子 (LaTeX 2ε), 112
\Bbb 数学公式中空心字体 (AMS-TeX), 99
\Bbbk $\Bbbk$ (AMS-TeX), 101
\because $\because$ (AMS-TeX), 103
\begin{...} 环境开始标记 (LaTeX 2ε), 114
\begin{document} 正文开始标志 (LaTeX 2ε), 106
\beta β, 27, 69
\beth $\beth$ (AMS-TeX), 101
\between $\between$ (AMS-TeX), 102
\bezier{n} 图形环境中绘制 Bézier 曲线的命令 (LaTeX 2ε), 149
\bf 正文中黑体字, 7
\biaosong 选择选择标宋字体的命令 (CCT), 177
\bibitem[...]{...}... 输入一项参考文件的结构 (LaTeX 2ε), 120
\bigcap $\bigcap$, 37
\bigcap $\bigcap$ $\bigcap$, 72
\bigcirc $\bigcirc$, 70
\bigcup $\bigcup$, 37
\bigcup $\bigcup$ $\bigcup$, 72
\Biggl 使左括号变最大, 37
\biggl 使左括号变大, 37
\Biggr 使右括号变最大, 37
\biggr 使右括号变大, 37
\Bigl 使左括号变很大, 37
\bigl 使左括号稍大一些, 36
\bigodot $\bigodot$ $\bigodot$, 72
\bigoplus $\bigoplus$ $\bigoplus$, 72
\bigotimes $\bigotimes$ $\bigotimes$, 72
\bigpagebreak 自然段大间距 (vanilla), 22
\Bigr 使右括号变很大, 37
\bigr 使右括号稍大一些, 36
\bigsqcup $\bigsqcup$ $\bigsqcup$, 72
\bigstar $\bigstar$ (AMS-TeX), 101
\bigtriangledown $\bigtriangledown$, 70
\bigtriangleup $\bigtriangleup$, 70
\biguplus $\biguplus$ $\biguplus$, 72
\bigvee $\bigvee$ $\bigvee$, 72
\bigwedge $\bigwedge$ $\bigwedge$, 72
\binom{...}{...} 二项式系数 (vanilla), 32
\blacklozenge $\blacklozenge$ (AMS-TeX), 101
\blacksquare $\blacksquare$ (AMS-TeX), 101
\blacktriangle $\blacktriangle$ (AMS-TeX), 101
\blacktriangledown $\blacktriangledown$ (AMS-TeX), 101

\blacktriangleleft ◀($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\blacktriangleright ▶($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\bold{...} 数学公式中的黑体字 (vanilla), 47
\book 参考文献书籍名称 (amsppt), 87
book 编写书的类 (LATEX 2ε), 107
\bot ⊥, 71
\botcaption 插图说明位于插图下面 (amsppt), 87
\bowtie ⋈, 70
\boxdot ⊡($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\boxminus ⊟($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\boxplus ⊞($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\boxtimes ⊠($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
bp 大点, 9
\breve 数学公式中的重音标记, 48
\bullet •, 70
\Bumpeq ≎($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\bumpeq ≏($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\by 参考文献中输入参考文件作者的结构 (amsppt), 87
\bye TEX 源文件结束, 10

\Cal 数学公式中花写体英文字母 (vanilla), 69
\Cal{...} 数学公式中的花写字体 (vanilla), 47
\cap ∩, 37, 70
\caption{...} 插入图表等的说明, 59
\captionwidth 插图说明宽度 (amsppt), 86
\cases...\\...\endcases 分段函数 (vanilla), 45
cc cicero, 9
cctart 中文论文类 (CCT), 176
cctbook 中文书类 (CCT), 176
\cdot ·, 70
\cdots 省略号, 47
\centerdot ·($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\chapter{...} 章标题 (LATEX 2ε), 116
\chapter{...}... 输入章编号和标题的结构 (amsppt), 88
\check 数学公式中的重音标记, 48
\chi χ, 69
\choose 数学公式中表示二项式系数 (LATEX 2ε), 133
\circ ∘, 70
\circeq ≗($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\circle*{d} 图形环境中画实心圆的命令 (LATEX 2ε), 147
\circle{d} 图形环境中画空心圆的命令 (LATEX 2ε), 147
\circlearrowleft ↺($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 104
\circlearrowright ↻($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 104
\circledast ⊛($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\circledcirc ⊚($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\circleddash ⊝($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\circledS Ⓢ($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\cite{...} 参考文献的引用结构 (LATEX 2ε), 120, 140
\cite{...} 正文中引用参考文献的结构, 88
\closing 信正文内容结束标志 (ltr), 52
\clubsuit ♣, 71
cm 厘米, 9
color 一种图形色彩管理宏程序包 (LATEX 2ε), 162
\columns 固定表格环中的列 (vanilla), 67
\columnsep{...} 双栏排版时确定两栏间距的结构 (LATEX 2ε), 107
\columnseprule 双栏排版时两栏间竖线的线宽度 (LATEX 2ε), 107

\complement ∁(AMS-TEX), 101
\cong ≅, 70
\coprod ∐ ∐, 72
\copyright ©, 7
\cos cos, 37, 38
\cosh cosh, 38
\cot cot, 38
\coth coth, 38
\cr 制表环境中的行结束符, 63
\csc csc, 38
\ctilde 数学公式中的重音标记, 48
ctxtops.bat 自动生成 .ps 文件的批处理文件 (LaTeX 2ε), 167
\cup ∪, 37, 70
\curlyeqprec ≼(AMS-TEX), 102
\curlyeqsucc ≽(AMS-TEX), 102
\curlyvee ⋎(AMS-TEX), 101
\curlywedge ⋏(AMS-TEX), 101
\curvearrowleft ↶(AMS-TEX), 104
\curvearrowright ↷(AMS-TEX), 104

\dag †, 7
\dagger †, 70
\daleth ℸ(AMS-TEX), 101
\dashboxl(w,h)[***]{...} 图形环境中绘制有虚线矩形框的命令 (LaTeX 2ε), 145
\dashv ⊣, 70
\date 表示写作日期的结构 (LaTeX 2ε), 115
\date 日期, 52
\date...\enddate 输入文章完成日期的结构 (amsppt), 85
\dbinom{...}{...} 单列公式尺寸二项式系数 (vanilla), 32
dd didôt 点, 9
\ddag ‡, 7
\ddagger ‡, 70
\ddot 数学公式中的重音标记, 48
\ddots 斜向省略号, 46
\dedicatory...\enddedicatory 输入作者写作目的的结构 (amsppt), 85
\def...{...} 自定义控制词, 57, 84
\def...#1#2...#n{...} 自定义带参量控制词, 58
\definition...\enddefinition 输入定义的结构 (amsppt), 86
\deg deg, 38
\Delta Δ, 69
\delta δ, 69
\demo{Proof}...\enddemo 证明输入结构 (vanilla), 21
description 一种列表环境 (LaTeX 2ε), 137
\det det, 38
\dfrac{...}{...} 输入单列公式尺寸分式结构 (vanilla), 32
\diagdown ⧹(AMS-TEX), 101
\diagup ⧸(AMS-TEX), 101
\diamond ⋄, 70
\diamondsuit ♢, 71
\digamma ϝ(AMS-TEX), 101
\dim dim, 38
displaymath 正文数学环境 (LaTeX 2ε), 123
\displaystyle 数学公式中单列公式尺寸符号 (LaTeX 2ε), 136
\div ÷, 70
\divideontimes ⋇(AMS-TEX), 102
\document 源文件正文部分开始标志 (AMS-TEX), 83
document 正文环境 (LaTeX 2ε), 114
\documentclass[...]{...} 选择类的结构 (LaTeX 2ε), 106
\documentstyle{...} 选用文件格式的命令 (AMS-TEX), 83

\dot 数学公式中的重音标记，48
\Doteq ≑(AMS-TEX)，102
\doteqdot ≑(AMS-TEX)，102
\dotfill 水平点填充，9
\dotplus ∔(AMS-TEX)，101
\dots ...，19
\dots 省略号（在 LaTeX 中可用在数学公式里），46
\doublebarwedge ⩞(AMS-TEX)，101
\doublecap ⋒(AMS-TEX)，101
\doublecup ⋓(AMS-TEX)，101
\Downarrow ⇓，71
\downarrow ↓，71
\downdownarrows ⇊(AMS-TEX)，104
\downharpoonleft ⇃(AMS-TEX)，104
\downharpoonright ⇂(AMS-TEX)，104
draft 类中不标识超宽行的选项 (LaTeX 2ε)，109
draft 类中用黑条标识超宽行的选项 (LaTeX 2ε)，109
\dsize 单列公式尺寸符号 (vanilla)，32，71
dvips 调入图形宏程序包的选项 (LaTeX 2ε)，162
dvips.exe 将 .dvi 格式文件转换成 PostScript 格式图形文件的程序 (LaTeX 2ε)，162
dvips32.exe 将 .dvi 格式文件转换成 PostScript 格式图形文件的程序 (LaTeX 2ε)，162

\ell ℓ，71
em 字母 M 的宽度，9
\email...\endemail 输入作者电子邮件地址的结构 (amsppt)，85
empty 一种页面格式 (LaTeX 2ε)，110
\emptyset ∅，70
\end{...} 环境结束标记 (LaTeX 2ε)，114
\end{document} 全文结束标志 (LaTeX 2ε)，106
\endcaption 插图说明结束标志 (amsppt)，86
\enddocument 源文件正文部分结束标志 (AMS-TEX)，83
\endgraf 在作者致谢结构中表示断行 (amsppt)，85
\endletter 信结尾标志 (ltr)，52
\endtopmatter 源文件标题部分结束标志 (AMS-TEX)，83
enumerate 一种列表环境 (LaTeX 2ε)，137
\enumi enumerate 环境中第一层标签编码计数器 (LaTeX 2ε)，139
\enumii enumerate 环境中第二层标签编码计数器 (LaTeX 2ε)，139
\enumiii enumerate 环境中第三层标签编码计数器 (LaTeX 2ε)，139
\enumiv enumerate 环境中第四层标签编码计数器 (LaTeX 2ε)，139
\epsilon ϵ，69
\eqcirc ≖(AMS-TEX)，102
eqnarray 自动编号多行单列公式环境 (LaTeX 2ε)，124
\eqslantgtr ⩾(AMS-TEX)，102
\eqslantless ⩽(AMS-TEX)，102
equation 自动编号单列数学公式环境 (LaTeX 2ε)，123
\equiv ≡，70
\eta η，69
\eth ð(AMS-TEX)，101
\evensidemargin 偶数页正文左边界的位置 (LaTeX 2ε)，111
ex 字母 x 的高度，9

\example...\endexample 输入例子的结构 (amsppt), 86
executivepaper 类的纸型选项 (LaTeX 2ε), 107
\exists ∃, 70
\exp exp, 38
\extracolsep{...} 表格环境中定义列间隔的结构 (LaTeX 2ε), 130

\fallingdotseq ≒(AMS-TeX), 102
\fangsong 选择仿宋字体的命令 (CCT), 177
\fill 空白填充, 9
\Finv ⅎ(AMS-TeX), 101
\flat ♭, 71
fleqn 类中单列公式对齐方式选项 (LaTeX 2ε), 108
\flushbottom 把正文版面底边安排在相同的位置上 (LaTeX 2ε), 112
\flushpar 自然段第一行不缩进 (vanilla), 22
\font...=*** scaled 定义新字体的结构, 72
\footnote 脚注, 21
\footnote"***"{...} 脚注 (vanilla), 22
\footskip 正文版面底部与页脚底部的距离 (LaTeX 2ε), 112
\forall ∀, 70
\format...\\ 矩阵列对齐方式 (vanilla), 45
\frac{...}{...} 输入分式结构, 32
\frak 数学公式中 fraktur 字体 (AMS-TeX), 99
\frame{...} 图形环境中的有框文本 (LaTeX 2ε), 149
\framebox(w,h)[***]{...} 图形环境中绘制有实线矩形框的命令 (LaTeX 2ε), 145
\frontmatter 书类中前言和目录部分的开始标志 (LaTeX 2ε), 122
\frown ⌢, 70

\Game ⅁(AMS-TeX), 101
\Gamma Γ, 27, 69
\gamma γ, 27, 69
\gcd gcd, 38
\ge ≥, 70
\geq ≥, 70
\geqq ≧(AMS-TeX), 102
\geqslant ⩾(AMS-TeX), 102
\gets ←, 71
\gg ≫, 70
\ggg ⋙(AMS-TeX), 102
\gggtr ⋙(AMS-TeX), 102
\gimel ℷ(AMS-TeX), 101
\gnapprox ⪊(AMS-TeX), 103
\gneq ⪈(AMS-TeX), 103
\gneqq ≩(AMS-TeX), 103
\gnsim ⋧(AMS-TeX), 103
graphics 一种图形宏程序包 (LaTeX 2ε), 162
\grave 数学公式中的重音标记, 48
\gtrapprox ⪆(AMS-TeX), 102
\gtrdot ⋗(AMS-TeX), 102
\gtreqless ⋛(AMS-TeX), 102
\gtreqqless ⪌(AMS-TeX), 102
\gtrless ≷(AMS-TeX), 102
\gtrsim ≳(AMS-TeX), 102
\gvertneqq ≩(AMS-TeX), 103

\halign to 设定表总宽的结构, 64
\halign to \hsize 设置表的总宽度等于正文版面宽的宽度, 65
\halign{...} 制表环境, 63, 168
\hat 数学公式中的重音标记, 48
\hbar ℏ, 70

\hbar $\hbar$($\mathcal{A}\mathcal{M}\mathcal{S}$-TEX), 101
\hcorrection{...} 调整正文版面水平位置 (vanilla), 60
\hdots 水平省略号 (vanilla), 46
\head...\endhead 输入标题的结构 (amsppt), 86
\headheight 页眉的高度 (LaTeX 2ε), 112
\headheight 页眉基线到版面顶部的距离 (LaTeX 2ε), 112
\heading...\endheading 节标题输入结构 (vanilla), 20
headings 一种页面格式 (LaTeX 2ε), 110
\heartsuit $\heartsuit$, 71
\heiti 选择黑体字的命令 (CCT), 177
\hfil 制表环境中的水平空白填充命令, 63
\hfill 水平空白填充, 9
\hidewith...\hidewith 表格环境中不以该项决定表格宽度, 66
\hline 表格环境中的横线 (LaTeX 2ε), 131
\hom $\hom$, 38
\hookleftarrow $\hookleftarrow$, 71
\hookrightarrow $\hookrightarrow$, 71
\hrulefill 水平线填充, 9
\hsize 表格环境中表示正文版面的宽度, 65
\hslash $\hbar$($\mathcal{A}\mathcal{M}\mathcal{S}$-TEX), 101
\idotsint $\int\cdots\int$ $\int\cdots\int$(vanilla), 72
\iff $\iff$, 71
\iiiint $\iiiint$(vanilla), 34
\iiiint $\iiiint$ $\iiiint$(vanilla), 72
\iiint $\iiint$(vanilla), 34
\iiint $\iiint$ $\iiint$(vanilla), 72
\iint $\iint$(vanilla), 34
\iint $\iint$ $\iint$(vanilla), 72
\Im $\Im$, 71
\imath $\imath$, 70
\impliedby $\impliedby$($\mathcal{A}\mathcal{M}\mathcal{S}$-TEX), 71
\implies $\implies$($\mathcal{A}\mathcal{M}\mathcal{S}$-TEX), 71
\in $\in$, 70
in 英寸, 9
\include{...} 把外部文件调入源文件中的命令 (LaTeX 2ε), 170
\includegraphics*[bx,by][ux,uy]{...} 在外部图形文件中剪裁一个矩形区域插入到正文中 (LaTeX 2ε), 162
\includegraphics[bx,by][ux,uy]{...} 将外部图形文件中图形插入到正文中 (LaTeX 2ε), 163
\includeonly{...,...,...} 决定所引入的文件中哪些需要重新编译 (LaTeX 2ε), 171
\indent 使自然段的第一行缩进 (LaTeX 2ε), 112
\index{...} 源文件中的索引记录命令 (LaTeX 2ε), 172
\inf $\inf$, 38
\infty ∞, 71
\input 读入外部文件, 18, 56
\int $\int$, 34
\int $\int$ $\int$, 72
\intercal $\intercal$($\mathcal{A}\mathcal{M}\mathcal{S}$-TEX), 102
\iota ι, 69
\it italic 体字, 5
\item 索引环境中主条目文本开始标记 (LaTeX 2ε), 174
\item 条目列表, 61
\itemitem 子条目列表 (vanilla), 61
itemize 一种列表环境 (LaTeX 2ε), 137
\jmath $\jmath$, 71

\jot 长度单位，41
\jour 参考文中输入参考文件所在献杂志名称的结构 (amsppt)，87

\kaishu 选择楷书字体的命令 (CCT)，177
\kappa κ，69
\ker ker，38
\key 参考文献的索引字符 (amsppt)，87
\keywords...\endkeywords 输入文章关键词的结构 (amsppt)，85

\L Ł，26
\l ł，26
\label{...} 表示自动编号的标签结构 (LaTeX 2ε)，116，140
\labelitemi enumerate 环境中第一层标签编号 (LaTeX 2ε)，139
\labelitemi itemize 环境中第一层标签 (LaTeX 2ε)，139
\labelitemii enumerate 环境中第二层标签编号 LaTeX 2ε)，139
\labelitemii itemize 环境中第二层标签 LaTeX 2ε)，139
\labelitemiii enumerate 环境中第三层标签编号 (LaTeX 2ε)，139
\labelitemiii itemize 环境中第三层标签 (LaTeX 2ε)，139
\labelitemiv enumerate 环境中第四层标签编号 (LaTeX 2ε)，139
\labelitemiv itemize 环境中第四层标签 (LaTeX 2ε)，139
\labrak [，69
\Lambda Λ，69
\lambda λ，69
\land $\land$，70
landscape 类的打印机进纸方向选项 (LaTeX 2ε)，107

\langle $\langle$，36，72
\LaTeX LaTeX，105
\LaTeXe LaTeX 2ε，116
latexidx.bat 能够自动创建索引表的批处理文件 (LaTeX 2ε)，175
\lbrack [，72
\lceil $\lceil$，72
\ldots 比较低的省略号，47
\le $\le$，70
\left 使左扩号成为可变尺寸符号 (必须与 \right 配合使用)，36
\left. 空的左括号，36
\Leftarrow $\Leftarrow$，71
\leftarrow $\leftarrow$，71
\leftarrowtail $\leftarrowtail$(AMS-TeX)，103
\leftharpoondown $\leftharpoondown$，71
\leftharpoonup $\leftharpoonup$，71
\leftleftarrows $\leftleftarrows$(AMS-TeX)，103
\Leftrightarrow $\Leftrightarrow$，71
\leftrightarrow $\leftrightarrow$，71
\leftrightarrows $\leftrightarrows$(AMS-TeX)，103
\leftrightharpoons $\leftrightharpoons$(AMS-TeX)，104
\leftrightsquigarrow $\leftrightsquigarrow$(AMS-TeX)，104
\leftthreetimes $\leftthreetimes$(AMS-TeX)，101
legalpaper 类的纸型选项 (LaTeX 2ε)，107
\leq $\leq$，70
leqno 类中单列公式编号安排方式选项 (LaTeX 2ε)，108
\leqq $\leqq$(AMS-TeX)，102
\leqslant $\leqslant$(AMS-TeX)，102
\lessapprox $\lessapprox$(AMS-TeX)，102
\lessdot $\lessdot$(AMS-TeX)，102
\lesseqgtr $\lesseqgtr$(AMS-TeX)，102
\lesseqqgtr $\lesseqqgtr$(AMS-TeX)，102
\lessgtr $\lessgtr$(AMS-TeX)，102

\lesssim $\lesssim$(AMS-TEX), 102
\letter 信正文内容开始标志 (ltr), 52
letter 编写信件的类 (LaTeX 2ε), 107
letterpaper 类的纸型选项 (LaTeX 2ε), 107
\lfloor $\lfloor$, 72
\lg lg, 38
\lim lim, 37, 38
\limits 输入积分区域的结构, 34
\limsup limsup, 38
\line(dx,xy){...} 图形环境中的画线命令 (LaTeX 2ε), 144
\linebreak 分行点, 24
\linethickness{...} 图形环境中修改粗线宽度的命令 (LaTeX 2ε), 150
\ll $\ll$, 70
\llcorner $\llcorner$(AMS-TEX), 100
\Lleftarrow $\Lleftarrow$(AMS-TEX), 103
\lll $\lll$(AMS-TEX), 102
\llless $\lll$(AMS-TEX), 102
\ln ln, 38
\lnapprox $\lnapprox$(AMS-TEX), 103
\lneq $\lneq$(AMS-TEX), 103
\lneqq $\lneqq$(AMS-TEX), 103
\lnot $\lnot$ 70
\lnsim $\lnsim$(AMS-TEX), 103
\loadbold 载入粗体字 (AMS-TEX), 84
\loadeufm 载入 AMS-TEX扩展 fraktur 字体 (AMS-TEX), 100
\loadmsbm 载入 AMS-TEX 空心字 (AMS-TEX), 100
\log log, 37, 38
\longleftarrow $\longleftarrow$, 71
\Longleftrightarrow $\Longleftrightarrow$, 71
\longleftrightarrow $\longleftrightarrow$, 71
\longmapsto $\longmapsto$, 71
\Longrightarrow $\Longrightarrow$, 71
\longrightarrow $\longrightarrow$, 71
\looparrowleft $\looparrowleft$(AMS-TEX), 103
\looparrowright $\looparrowright$(AMS-TEX), 103
\lozenge $\lozenge$(AMS-TEX), 101
\lrcorner $\lrcorner$(AMS-TEX), 100
\Lsh $\Lsh$(AMS-TEX), 104
\ltimes $\ltimes$(AMS-TEX), 101
\lvertneqq $\lvertneqq$(AMS-TEX), 103

\magnification=... 确定版面的整体放大倍数 (vanilla), 62
\magstep0 版面放大因子 1.000, 62
\magstep1 版面放大因子 1.2, 62
\magstep2 版面放大因子 1.44, 62
\magstep3 版面放大因子 1.728, 62
\magstep4 版面放大因子 2.074, 62
\magstep5 版面放大因子 2.488, 62
\magstephalf 版面放大因子 1.095, 62
\mainmatter 书类中正文部分开始的标志 (LaTeX 2ε), 122
\makebox(w,h)[...]{...} 图形环境中绘制无框矩形的命令 (LaTeX 2ε), 145
makeidx.sty 实现自动建立索引功能的宏程序包 (LaTeX 2ε), 175
\makeindex 在编译源文件时生成索引文件 .idx(LaTeX 2ε), 173
makeindx.exe 将索引文件 .idx 转换成索引环境文件 .ind 的程序 (LaTeX 2ε), 174
\maketitle 确认文章标题作者和日期等的输入 (LaTeX 2ε), 115
\mapsto $\mapsto$, 71
\markboth{...}{...} 定义左右页眉的结构 (LaTeX 2ε), 110
\markright{...} 定义右页眉的结构 (LaTeX 2ε), 110
math 正文数学环境 (LaTeX 2ε), 114, 123

\mathbf{...} 数学公式中输入粗字体 (LaTeX 2ε), 135
\mathbreak 数学公式分行点 (vanilla), 29
\mathcal 数学公式中表示其后面的一个字母为花写体英文字母 (LaTeX 2ε), 69
\mathcal{...} 数学公式中输入花写字体 (LaTeX 2ε), 135
\mathindent 决定单列公式左所进量 (LaTeX 2ε), 108
\mathit{...} 数学公式中输入 italic 字体 (LaTeX 2ε), 135
\mathrm{...} 数学公式中输入 roman 字体 (LaTeX 2ε), 135
\mathsf{...} 数学公式中输入 slanted 字体 (LaTeX 2ε), 135
\mathtt{...} 数学公式中输入打字机字体 (LaTeX 2ε), 135
\matrix...\endmatrix 数组排列结构 (vanilla), 43
\max max, 37, 38
\mbox{...} 数学公式中输入文本的结构 (LaTeX 2ε), 135
\measuredangle ∡(AMS-TeX), 101
\medpagebreak 自然段中间距 (vanilla), 22
\mho ℧(AMS-TeX), 101
\mid ∣, 70
\midinsert 在页面中部插图 (amsppt), 86
\midspace{...} 插入垂直空间 (vanilla), 59
\min min, 38
mm 毫米, 9
\models ⊨, 70
\Monograph 编排单独一章的内容 (AMS-TeX), 84
\Monograph 说明源文件是某书的一部分 (amsppt), 88
\mp ∓, 70
\mu μ, 69
\multimap ⊸(AMS-TeX), 104
\multiput(x,y)(dx,dy){n}{...} 图形环境中的图形功能重复设置结构 (LaTeX 2ε), 142
\multispan5 表格环境中占五个列的项, 66
myheadings 一种页面格式 (LaTeX 2ε), 110

\nabla ∇, 70
\narrower 稍窄于正文版面的宽度排版 (vanilla), 61
\natural ♮, 71
\ncong ≇(AMS-TeX), 103
\ne ≠, 70
\nearrow ↗, 71
\neg ¬, 70
\neq ≠, 70
\newline 另起一行, 24
\newpage 另起一页, 59
\newsavebox{~~~} 图形环境中定义子图形名的结构 (LaTeX 2ε), 152
\newsymbol... 定义数学符号表达式 (AMS-TeX), 100
\nexists ∄(AMS-TeX), 101
\ngeq ≱(AMS-TeX), 103
\ngeqq ≱(AMS-TeX), 103
\ngeqslant ≱(AMS-TeX), 103
\ngtr ≯(AMS-TeX), 103
\ni ∋, 70
\nLeftarrow ⇍(AMS-TeX), 104
\nleftarrow ↚(AMS-TeX), 104
\nLeftrightarrow ⇎(AMS-TeX), 104
\nleftrightarrow ↮(AMS-TeX), 104

\nleq $\not\leq$(AMS-TEX), 103
\nleqq $\not\leqq$(AMS-TEX), 103
\nleqslant $\not\leqslant$(AMS-TEX), 103
\nless $\not<$(AMS-TEX), 103
\nmid $\nmid$(AMS-TEX), 103
\no 参考文献中输入参考文献编号的结构 (amsppt), 87
\noalign{\hrule} 表格环境中的水平横线, 66
\noalign{\smallskip} 表格环境中留出竖直方向间隙, 67
\nofrills{...} 修改章标题文字及编号 (amsppt), 88
\noindent 使自然段的第一行不缩进 (LaTeX 2ε), 112
\nolinebreak 禁止分行点, 59
\nonumber 取消公式自动编号 (LaTeX 2ε), 125
\nopagebreak 禁止分页点, 59
\nopagenumbers 不打印页码 (vanilla), 61, 84
\NoRunningHeads 取消页眉, 84
\not 否定数学关系符号, 70
\notin $\notin$, 70
notitlepage 类中标题页安排方式选项 (LaTeX 2ε), 108
\nparallel $\nparallel$(AMS-TEX), 103
\nprec $\not\prec$(AMS-TEX), 103
\npreceq $\not\preceq$(AMS-TEX), 103
\nRightarrow $\nRightarrow$(AMS-TEX), 104
\nrightarrow $\nrightarrow$(AMS-TEX), 104
\nshortmid $\nshortmid$(AMS-TEX), 103
\nshortparallel $\nshortparallel$(AMS-TEX), 103
\nsim $\nsim$(AMS-TEX), 103
\nsubseteq $\not\subseteq$(AMS-TEX), 103
\nsubseteqq $\not\subseteqq$(AMS-TEX), 103
\nsucc $\not\succ$(AMS-TEX), 103
\nsucceq $\not\succeq$(AMS-TEX), 103
\nsupseteq $\not\supseteq$(AMS-TEX), 103
\nsupseteqq $\not\supseteqq$(AMS-TEX), 103
\ntriangleleft $\not\triangleleft$(AMS-TEX), 103
\ntrianglelefteq $\not\trianglelefteq$(AMS-TEX), 103
\ntriangleright $\not\triangleright$(AMS-TEX), 103
\ntrianglerighteq $\not\trianglerighteq$(AMS-TEX), 103
\nu ν, 69
\nVDash $\nVDash$(AMS-TEX), 103
\nVdash $\nVdash$(AMS-TEX), 103
\nvDash $\nvDash$(AMS-TEX), 103
\nvdash $\nvdash$(AMS-TEX), 103
\nwarrow $\nwarrow$, 71

\O $\emptyset$, 26
\o $\o$, 26
\oddsidemargin 奇数页正文左边界的位置 (LaTeX 2ε), 111
\odot $\odot$, 70
\OE Œ, 26
\oe œ, 26
\offinterlineskip 表格环境各个行之间的距离, 65
\oint $\oint$, 34
\oint $\oint$ $\oint$, 72
\Omega Ω, 69
\omega ω, 35, 69
\ominus $\ominus$, 70
\omit 表格环境中摆脱序行限制的项, 66
onecolumn 类的单栏排版选项 (LaTeX 2ε), 107
oneside 类的页码位置选项 (LaTeX 2ε), 108
openany 类中新一章页码安排方式选项 (LaTeX 2ε), 108
openbib 类中确定参考文献排列方式的选项 (LaTeX 2ε), 109

openright 类中新一章页码安排方式选项 (LaTeX 2ε), 108
\oplus ⊕, 70
\oslash ⊘, 70
\otimes ⊗, 70
\oval(w,h)[...] 图形环境中绘制圆角矩形的命令 (LaTeX 2ε), 147
\overline 在数学公式上面划线, 35
\overrightarrow 在数学公式上面划箭头线, 35
\owns ∋, 70

\P ¶, 7
\pagebreak 分页点, 59
\pageheight{...} 定义正文版面高度 (vanilla), 60, 84
\pageno 设置首页页码 (vanilla), 61, 84
\pagenumbering{...} 定义页码编号方式的结构 (LaTeX 2ε), 111
\pageref{...} 标签所在页码的引用结构 (LaTeX 2ε), 140
\pagerheight 纸张的高度 (LaTeX 2ε), 112
\pagers 参考文献中输入参考文件所在页码的结构 (amsppt), 87
\pagestyle{...} 修改全文页面格式 (LaTeX 2ε), 109
\pagewidth{...} 定义正文版面宽度 (vanilla), 60, 84
\paper 参考文献中输入论文标题的结构 (amsppt), 87
\paperwidth 纸张的宽度 (LaTeX 2ε), 112
\par 结束自然段, 7
\paragraph{...} 段标题 (LaTeX 2ε), 116
\parallel ∥, 70
\parbox[...]{...}{...} 图形环境中的多行文本的输入结构 (LaTeX 2ε), 143
\parindent 自然段第一行起始位置的缩进量 (LaTeX 2ε), 111
\parskip 两个自然段落之间的距离 (LaTeX 2ε), 111
\part{...} 部分标题 (LaTeX 2ε), 116
\partial ∂, 35, 71
patchdvi.exe 将含中文的 .dvi 转换成标准的英文格式 .dvi 格式的软件, 167
pc pica, 9
\perp ⊥, 70
\Phi Φ, 69
\phi ϕ, 69
\Pi Π, 69
\pi π, 69
picture 图形环境 (LaTeX 2ε), 141
\pitchfork ⋔(AMS-TeX), 102
plain 一种页面格式 (LaTeX 2ε), 110
\pm ±, 70
\pmatrix...\endpmatrix 矩阵结构 (vanilla), 43
\Pr Pr, 38
\prec ≺, 70
\precapprox ⪷(AMS-TeX), 102
\preccurlyeq ≼(AMS-TeX), 102
\preceq ⪯, 70
\precnapprox ⪹(AMS-TeX), 103
\precneqq ⪵(AMS-TeX), 103
\precnsim ⋨(AMS-TeX), 103
\precsim ≾(AMS-TeX), 102
\prime ′, 31, 70
\printindex makeidx 宏程序包中表示索引表的命令 (LaTeX 2ε), 175
\proclaim{***}...\endproclaim 定理输入结构 (vanilla), 21
\prod ∏ $\prod$, 72

\propto ∝, 70
\protect 使控制词能够正确传递到目录表、页眉等地方 (LaTeX 2ε), 119
\Psi Ψ, 69
\psi ψ, 69
pt 点, 9
\put(x,y){...} 图形环境中的图形功能设置结构 (LaTeX 2ε), 142

\qbezier[n]... 图形环境中绘制 Bézier 曲线的命令 (LaTeX 2ε), 149
\qquad 四个英文字母宽的空格, 39
\quad 两个英文字母宽的空格, 39

\raggedbottom 使正文版面底边位置的安排稍有变化 (LaTeX 2ε), 112
\raggedright 将正文的右边界打印成不规则形状 (vanilla), 61
\rangle ⟩, 36, 72
\rbrack ⌋, 72
\rbrack], 69
\rceil ⌉, 72
\Re ℜ, 71
\ref...\endref 输入一条参考文献的结构 (amsppt), 87
\ref{...} 自动编号的引用结构 (LaTeX 2ε), 140
\reflectbox{...} 图形的水平反射命令 (LaTeX 2ε), 165
\Refs 参考文件标题 (amsppt), 87
\remark...\endremark 输入注解的结构 (amsppt), 86
\renewcommand{...}{***} 重新定义已有控制词的结构 (LaTeX 2ε), 139
report 编写学术报告的类 (LaTeX 2ε), 107
\resizebox{!}{h}{...} 在保持原图形长宽比例的条件下将图形的高缩放到给定尺寸的命令 (LaTeX 2ε), 164
\resizebox{w}{!}{...} 在保持原图形长宽比例的条件下将图形的宽缩放到给定尺寸的命令 (LaTeX 2ε), 164
\resizebox{w}{h}{...} 图形缩放的命令 (LaTeX 2ε), 163
\restriction ↾(AMS-TeX), 104
\returnaddress 回信地址 (ltr), 52
\rfloor ⌋, 72
\rho ρ, 69
\right 使右扩号成为可变尺寸符号 (必须与 \left 配合使用), 36
\Rightarrow ⇒, 71
\rightarrow →, 71
\rightarrowtail ↣(AMS-TeX), 103
\rightharpoondown ⇁, 71
\rightharpoonup ⇀, 71
\rightleftarrows ⇄(AMS-TeX), 103
\rightleftharpoons ⇌, 71
\rightleftharpoons ⇌(AMS-TeX), 104
\rightrightarrows ⇉(AMS-TeX), 103
\rightsquigarrow ⇝(AMS-TeX), 104
\rightthreetimes ⋌(AMS-TeX), 101
\risingdotseq ≓(AMS-TeX), 102
\rith. 空的右括号, 36
\rlap 表格环境中其后的字符对计算列宽不起作用, 67
\rm roman 体字, 6
\Roman 大写罗马数字编号 (LaTeX 2ε), 139
\roman 小写罗马数字编号 (LaTeX 2ε), 139
\root...\of... 开方号, 35
\rotatebox{a}{...} 将图形按逆时针方向旋转到给定角度的命令 (LaTeX 2ε), 166
\Rrightarrow ⇛(AMS-TeX), 103
\Rsh ↱(AMS-TeX), 104

\rtimes ⋊($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101

\S §, 7
\savebox{~~~}(w,h)[***]{...} 图形环境中存储子图形的结构 (LaTeX 2ε), 152
\sc 大写英文字体, 19
\scalebox{w}[h]{...} 图形的长和宽按照给定比例缩放的命令 (LaTeX 2ε), 164
\scriptscriptstyle 数学公式中最底层下标尺寸符号 (LaTeX 2ε), 136
\scriptstyle 数学公式中下标尺寸符号 (LaTeX 2ε), 136
\searrow ↘, 71
\sec sec, 38
\section{...} 节标题 (LaTeX 2ε), 116
|see 索引条目记录中命令 (LaTeX 2ε), 173
\setcounter{page}{...} 设置当前页码数值的结构 (LaTeX 2ε), 111
\setcounter{scnumdepth}{...} 设定计数器深度的结构 (LaTeX 2ε), 118
\setcounter{tocdepth}{...} 设立目录表中所包含标题范围的结构 (LaTeX 2ε), 118
\setlength{...}{...} 设置长度单位所代表长度值的结构 (LaTeX 2ε), 113
\setlength{\mathindent}{...} 定义单列公式左对齐位置的结构 (LaTeX 2ε), 123
\setminus \, 70
\settabs 固定列宽表格环境, 67
\settowidth{...}{...} 设置长度单位所代表长度的结构 (LaTeX 2ε), 113
\sharp ♯, 71
\shortmid ∣($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\shortparallel ∥($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\shortstack[...]{***} 图形环境中竖直排放文本的命令 (LaTeX 2ε), 148
\Sigma Σ, 69
\sigma σ, 69
\sim ∼, 70
\simeq ≃, 70
\sin sin, 37, 38
\sinh sinh, 38
\sl sland 字体, 6
\smallfrown ⌢($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\smallint ∫, 71
\smallpagebreak 自然段小间距 (vanilla), 22
\smallsetminus ∖($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\smallsmile ⌣($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\smile ⌣, 70
\songti 选择宋体字的命令 (CCT), 177
\Sp...\\...\endSp 输入多行下标结构 (vanilla), 34
\spadesuit ♠, 71
\special{em:lineto} 图形环境中画线命令 (emtex), 153
\special{em:linewidth w} 图形环境中设置线宽的命令 (emtex), 153
\special{em:moveto} 图形环境中设置当前点坐标的命令 (emtex), 153
\specialhead...\endspecialhead 输入标题的结构 (amsppt), 86
\sphericalangle ∢($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\sqcap ⊓, 70
\sqcup ⊔, 70
\sqrt 根号, 35
\sqsubset ⊏($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\sqsubseteq ⊑, 70
\sqsupset ⊐($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 102
\sqsupseteq ⊒, 70
\square □($\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX), 101
\ss ß, 26

\star ⋆, 70
\strut 表格环境中一行的高度, 66
\subheading{...} 子标题输入结构 (vanilla), 20
\subitem 索引环境中子条目文本开始标记 (LaTeX 2ε), 174
\subjclass...\endsubjclass 输入文章分类号的结构 (amsppt), 85
\subparagraph{...} 小段标题 (LaTeX 2ε), 116
\subsection{...} 小节标题 (LaTeX 2ε), 116
\Subset ⋐(AMS-TeX), 102
\subset ⊂, 70
\subseteq ⊆, 70
\subseteqq ⫅(AMS-TeX), 102
\subsetneq ⊊(AMS-TeX), 103
\subsetneqq ⫋(AMS-TeX), 103
\subsubitem 索引环境中子子条目文本开始标记 (LaTeX 2ε), 174
\subsubsection{...} 子节标题 (LaTeX 2ε), 116
\succ ≻, 70
\succapprox ⪸(AMS-TeX), 102
\succcurlyeq ≽(AMS-TeX), 102
\succeq ⪰, 70
\succnapprox ⪺(AMS-TeX), 103
\succneqq ⪶(AMS-TeX), 103
\succnsim ⋩(AMS-TeX), 103
\succsim ≿(AMS-TeX), 102
\sum ∑, 33
\sum $\sum$, 72
\sup sup, 38
\Supset ⋑(AMS-TeX), 102
\supset ⊃, 70
\supseteq ⊇, 70
\supseteqq ⫆(AMS-TeX), 102
\supsetneq ⊋(AMS-TeX), 103
\supsetneqq ⫌(AMS-TeX), 103
\surd √, 71
\swarrow ↙, 71
\syntax 编译源文件时阻止生成 .dvi 文件 (vanila), 55

\tableofcontents 自动建立目录表的命令 (LaTeX 2ε), 118
\tabskip 表格环境中表示各个列间隔的距离, 64
\tabskip= 表格环境中定义各个列间隔的距离, 65
tabular 表格环境 (LaTeX 2ε), 129
\tag 单列公式的编号 (vanilla), 39
\TagsOnRight 单列公式的编号打印在右边 (vanilla), 59
\tan tan, 38
\tanh tanh, 38
\tau τ, 69
\tbinom{...}{...} 正文公式尺寸二项式系数 (vanilla), 32
\TeX TeX, 9
\text{...} 公式中打印文本 (vanilla), 38
\textheight 正文的高度, 即版面的高度 (LaTeX 2ε), 111
\textstyle 数学公式中文本公式尺寸符号 (LaTeX 2ε), 136
\textwidth 正文的宽度, 即版面的宽度 (LaTeX 2ε), 111
\tfrac{...}{...} 输入正文公式尺寸分式结构 (vanilla), 32
\thanks...\endthanks 输入作者致谢的结构 (amsppt), 85
\thanks{...} 表示感谢、资助等的结构 (LaTeX 2ε), 116
thebibliography 参考文献环境 (LaTeX 2ε), 120

theindex 索引环境 (LaTeX 2ε), 174
\therefore ∴(AMS-TeX), 103
\Theta Θ, 69
\theta θ, 69
\thickapprox ≈(AMS-TeX), 102
\thicklines 绘图环境中选择粗线的命令 (LaTeX 2ε), 150
\thicksim ∼(AMS-TeX), 102
\thinlines 绘图环境中选择细线的命令 (LaTeX 2ε), 150
\thispagestyle{...} 修改当前页的页面格式 (LaTeX 2ε), 110
\times ×, 70
\title...\endtitle 标题输入结构 (vanilla), 19, 85
\title...\page{...}\endtitle 输入目录项的结构 (amsppt), 89
\title{...} 文章标题输入结构 (LaTeX 2ε), 115
titlepage 类中标题页安排方式选项 (LaTeX 2ε), 108
\to →, 71
\toc...\endtoc 输入目录的结构 (amsppt), 89
\today 计算机时钟时间, 52
\top ⊤, 71
\topcaption 插图说明位于插图上面 (amsppt), 86
\topinsert 在页面顶部插图 (amsppt), 86
\topmargin 页眉顶部的位置 (LaTeX 2ε), 112
\topmatter 源文件标题部分开始标志 (AMS-TeX), 83
\topskip 从版面顶部到正文第一行基线的距离 (LaTeX 2ε), 112
\translator...\endtranslator 输入译者姓名的结构 (amsppt), 85
\triangle △, 71
\triangledown ▽(AMS-TeX), 101
\triangleleft ◁, 70
\trianglelefteq ⊴(AMS-TeX), 102
\triangleq ≜(AMS-TeX), 102
\triangleright ▷, 70
\trianglerighteq ⊵(AMS-TeX), 102
\tsize 正文公式尺寸符号 (vanilla), 32, 71
twocolumn 类的双栏排版选项 (LaTeX 2ε), 107
\twoheadleftarrow ↞(AMS-TeX), 103
\twoheadrightarrow ↠(AMS-TeX), 103
twoside 类的页码位置选项 (LaTeX 2ε), 108

\ulcorner ⌜(AMS-TeX), 100
\underbar 下划线, 3
\underline 在数学公式下面划线, 35
\unitlength 表示图形环境中的单位坐标长 (LaTeX 2ε), 141
\Uparrow ⇑, 71
\uparrow ↑, 71
\Updownarrow ⇕, 71
\updownarrow ↕, 71
\upharpoonleft ↿(AMS-TeX), 104
\upharpoonright ↾(AMS-TeX), 104
\uplus ⊎, 70
\Upsilon Υ, 69
\upsilon υ, 69
\upuparrows ⇈(AMS-TeX), 104
\urcorner ⌝(AMS-TeX), 100
\UseAMSsymbols 载入 AMS-TeX 扩充的数学符号, 100
\usebox{~~~ 图形环境中调用子图形的命令 (LaTeX 2ε), 152

\usepackage[...]{...} 载入宏程序包的结构 (LaTeX 2ε), 109
\usepackage[dvips]{graphics,color} 调入图形功能宏程序包 (LaTeX 2ε), 162

\varepsilon ε, 69
\varkappa $\varkappa$(AMS-TeX), 101
\varnothing $\varnothing$(AMS-TeX), 101
\varphi φ, 69
\varpi ϖ, 69
\varpropto $\varpropto$(AMS-TeX), 102
\varrho ϱ, 69
\varsigma ς, 69
\varsubsetneq $\varsubsetneq$(AMS-TeX), 103
\varsubsetneqq $\varsubsetneqq$(AMS-TeX), 103
\varsupsetneq $\varsupsetneq$(AMS-TeX), 103
\varsupsetneqq $\varsupsetneqq$(AMS-TeX), 103
\vartheta ϑ, 69
\vartriangle $\vartriangle$(AMS-TeX), 101
\vartriangleleft $\vartriangleleft$(AMS-TeX), 102
\vartriangleright $\vartriangleright$(AMS-TeX), 102
\vbox 数学公式中的正文环境, 63
\vcorrection{...} 调整正文版面垂直位置 (vanilla), 60
\Vdash $\Vdash$(AMS-TeX), 102
\vDash $\vDash$(AMS-TeX), 102
\vdash $\vdash$, 70
\vdots 垂直省略号, 46
\vec 数学公式中的重音标记, 48
\vector(dx,xy){...} 图形环境中的绘制箭头线的命令 (LaTeX 2ε), 145
\vee $\vee$, 70
\veebar $\veebar$(AMS-TeX), 101
\Vert $\Vert$, 71, 72
\vert $\vert$, 69, 72
\vfill 垂直空白填充, 9
\vglue 设置空白, 18
\vol 参考文献中杂志卷号 (amsppt), 87
\vrule 表格环境中高度等于行高度的垂线, 66
\vskip{...} 垂直空白, 22
\vspace{...} 垂直空隙, 41
\vtop 表格环境中的一种列排版方式 (vanilla), 65
\Vvdash $\Vvdash$(AMS-TeX), 102

\wedge $\wedge$, 70
\widehat 数学公式中的重音标记, 48
\widestnumber{...} 说明参考文献的最长索引字符 (amsppt), 87
\widetilde 数学公式中的重音标记, 48
\wp $\wp$, 71
\wr $\wr$, 70

\Xi Ξ, 69
\xi ξ, 69

\yr 参考文献中输入参考文件所在杂志年份的结构 (amsppt), 87

\zeta ζ, 69
\zihao{n} 中文字号设置命令 (CCT), 177
\ziti{...} 选择字体的命令 (CCT), 177

Document generated by Anna’s Archive around 2023-2024 as part of the DuXiu collection (https://annas-blog.org/duxiu-exclusive.html).

Images have been losslessly embedded. Information about the original file can be found in PDF attachments. Some stats (more in the PDF attachments):

{
  "filename": "MTA0MjU0NDYuemlw",
  "filename_decoded": "10425446.zip",
  "filesize": 16385147,
  "md5": "edd7607cb50dccbf28aad93ad1ef20b5",
  "header_md5": "b209ec434f34cfba5697e712ab46f6ec",
  "sha1": "2370799b30305a69cd4a32ea4d48681a10b440bf",
  "sha256": "b247767c07bbce4e294119f5454d346b51727dae11ba9ba7f217fb0666123b06",
  "crc32": 4143114987,
  "zip_password": "52gv",
  "uncompressed_size": 16736694,
  "pdg_dir_name": "TEX\u3001AMS\u2014\u2014TEX\u548cLATEX\u4f7f\u7528\u7b80\u4ecb_10425446",
  "pdg_main_pages_found": 217,
  "pdg_main_pages_max": 217,
  "total_pages": 231,
  "total_pixels": 1638821889,
  "pdf_generation_missing_pages": false
}