

# GB/T 7714—2015 BibT<sub>E</sub>X style

Zeping Lee\*

2021/06/20      v2.1.2

## 摘要

The gbt7714 package provides a BibT<sub>E</sub>X implementation for the China's bibliography style standard GB/T 7714—2015. It consists of two bst files for numerical and authoryear styles as well as a L<sup>A</sup>T<sub>E</sub>X package which provides the citation style defined in the standard. It is compatible with natbib and supports language detection (Chinese and English) for each bibliography entry.

## 1 简介

GB/T 7714—2015 《信息与文献 参考文献著录规则》<sup>[1]</sup>（以下简称“国标”）是中国的参考文献推荐标准。本宏包是国标的 BibT<sub>E</sub>X<sup>[2]</sup> 实现，具有以下特性：

- 兼容 natbib 宏包<sup>[3]</sup>
- 支持顺序编码制和著者-出版年制两种风格
- 自动识别语言并进行相应处理
- 提供了简单的接口供用户修改样式

本宏包的主页：<https://github.com/CTeX-org/gbt7714-bibtex-style>。

## 2 版本 v2.0 的重要修改

从 v2.0 版本开始(2020-03-04),用户必须在文档中使用 `\bibliographystyle` 命令选择参考文献样式,如 `gbt7714-numerical` 或 `gbt7714-author-year`。在早期的版本中,选择文献样式的方法是将 `numbers` 或 `super` 等参数传递给 `gbt7714`,而不能使用 `\bibliographystyle`。这跟标准的 L<sup>A</sup>T<sub>E</sub>X 接口不一致,所以将被弃用。

---

\*zepinglee AT gmail.com

### 3 使用方法

按照国标的规定，参考文献的标注体系分为“顺序编码制”和“著者-出版年制”。用户应在导言区调用宏包 `gbt7714`，并且使用 `\bibliographystyle` 命令选择参考文献表的样式，比如：

```
\bibliographystyle{gbt7714-numerical} % 顺序编码制
```

或者

```
\bibliographystyle{gbt7714-author-year} % 著者-出版年制
```

注意，版本 v2.0 更改了设置参考文献表样式的方法，要求直接使用 `\bibliographystyle`，不再使用宏包的参数，而且更改了 `bst` 的文件名。

顺序编码制的引用标注默认使用角标式，如“张三<sup>[2]</sup>提出”。如果要使用正文模式，如“文献 [3] 中说明”，可以使用 `\citestyle` 命令进行切换：

```
\citestyle{numbers}
```

同一处引用多篇文献时，应当将各篇文献的 `key` 一同写在 `\cite` 命令中。如遇连续编号，默认会自动转为起讫序号并用短横线连接（见 `natbib` 的 `compress` 选项）。如果要对引用的编号进行自动排序，需要在调用 `gbt7714` 时加 `sort&compress` 参数：

```
\usepackage[sort&compress]{gbt7714}
```

这些参数会传给 `natbib` 处理。

若需要标出引文的页码，可以标在 `\cite` 的可选参数中，如 `\cite[42]{knuth84}`。更多的引用标注方法可以参考 `natbib` 宏包的使用说明<sup>[3]</sup>。

使用时需要注意以下几点：

- `.bib` 数据库应使用 UTF-8 编码。
- 使用著者-出版年制参考文献表时，中文的文献必须在 `key` 域填写作者姓名的拼音，才能按照拼音排序，详见第 6 节。

### 4 文献类型

国标中规定了 16 种参考文献类型，表 1 列举了 `bib` 数据库中对应的文献类型。这些尽可能兼容 `BibTeX` 的标准类型，但是新增了若干文献类型（带 \* 号）。

表 1: 全部文献类型

文献类型	标识代码	Entry Type
普通图书	M	book
图书的析出文献	M	incollection
会议录	C	proceedings
会议录的析出文献	C	inproceedings 或 conference
汇编	G	collection*
报纸	N	newspaper*
期刊的析出文献	J	article
学位论文	D	mastersthesis 或 phdthesis
报告	R	techreport
标准	S	standard*
专利	P	patent*
数据库	DB	database*
计算机程序	CP	software*
电子公告	EB	online*
档案	A	archive*
舆图	CM	map*
数据集	DS	dataset*
其他	Z	misc

## 5 著录项目

由于国标中规定的著录项目多于 Bib<sub>T</sub>E<sub>X</sub> 的标准域，必须新增一些著录项目（带 \* 号），这些新增的类型在设计时参考了 BibLa<sub>T</sub>E<sub>X</sub>，如 date 和 urldate。本宏包支持的全部域如下：

**author** 主要责任者

**title** 题名

**mark\*** 文献类型标识

**medium\*** 载体类型标识

**translator\*** 译者

**editor** 编辑

**organization** 组织（用于会议）

**booktitle** 图书题名

**series** 系列

**journal** 期刊题名

**edition** 版本

**address** 出版地

**publisher** 出版者  
**school** 学校（用于 phdthesis）  
**institution** 机构（用于 techreport）  
**year** 出版年  
**volume** 卷  
**number** 期（或者专利号）  
**pages** 引文页码  
**date\*** 更新或修改日期  
**urldate\*** 引用日期  
**url** 获取和访问路径  
**doi** 数字对象唯一标识符  
**langid\*** 语言  
**key** 拼音（用于排序）

不支持的 Bib<sub>T</sub><sub>E</sub>X 标准著录项目有 `annotate`, `chapter`, `crossref`, `month`, `type`。

本宏包默认情况下可以自动识别文献语言，并自动处理文献类型和载体类型标识，但是在少数情况下需要用户手动指定，如：

```
@misc{citekey,  
  langid = {japanese},  
  mark   = {Z},  
  medium = {DK},  
  ...  
}
```

可选的语言有 `english`, `chinese`, `japanese`, `russian`。

## 6 文献列表的排序

国标规定参考文献表采用著者-出版年制组织时，各篇文献首先按文种集中，然后按著者字顺和出版年排列；中文文献可以按著者汉语拼音字顺排列，也可以按著者的笔画笔顺排列。然而由于 Bib<sub>T</sub><sub>E</sub>X 功能的局限性，无法自动获取著者姓名的拼音或笔画笔顺，所以必须在 `bib` 数据库中的 `key` 域手动录入著者姓名的拼音，如：

```
@book{capital,  
  author = {马克思 and 恩格斯},  
  key    = {ma3 ke4 si1 & en1 ge2 si1},  
  ...  
}
```

## 7 自定义样式

BibTeX 对自定义样式的支持比较有限，所以用户只能通过修改 `bst` 文件来修改文献列表的格式。本宏包提供了一些接口供用户更方便地修改。

在 `bst` 文件开始处的 `load.config` 函数中，有一组配置参数用来控制样式，表 2 列出了每一项的默认值和功能。若变量被设为 `#1` 则表示该项被启用，设为 `#0` 则不启用。默认的值是严格遵循国标的配置。

表 2: 参考文献表样式的配置参数

参数值	默认值	功能
<code>uppercase.name</code>	<code>#1</code>	将著者姓名转为大写
<code>max.num.authors</code>	<code>#3</code>	输出著者的最多数目
<code>year.after.author</code>	<code>#0</code>	年份置于著者之后
<code>period.after.author</code>	<code>#0</code>	著者和年份之间使用句点连接
<code>italic.book.title</code>	<code>#0</code>	西文书籍名使用斜体
<code>sentence.case.title</code>	<code>#1</code>	将西文的题名转为 sentence case
<code>link.title</code>	<code>#0</code>	在题名上添加 url 的超链接
<code>title.in.journal</code>	<code>#1</code>	期刊是否显示标题
<code>space.before.mark</code>	<code>#0</code>	文献类型标识前是否有空格
<code>show.mark</code>	<code>#1</code>	显示文献类型标识
<code>show.medium.type</code>	<code>#1</code>	显示载体类型标识
<code>italic.journal</code>	<code>#0</code>	西文期刊名使用斜体
<code>show.missing.address.publisher</code>	<code>#0</code>	出版项缺失时显示“出版者不详”
<code>space.before.pages</code>	<code>#1</code>	页码与前面的冒号之间有空格
<code>only.start.page</code>	<code>#0</code>	只显示起始页码
<code>wave.dash.in.pages</code>	<code>#0</code>	起止页码使用波浪号
<code>show.urldate</code>	<code>#1</code>	显示引用日期 <code>urldate</code>
<code>show.url</code>	<code>#1</code>	显示 url
<code>show.doi</code>	<code>#1</code>	显示 DOI
<code>show.preprint</code>	<code>#0</code>	显示预印本信息
<code>show.note</code>	<code>#0</code>	显示 note 域的信息
<code>end.with.period</code>	<code>#1</code>	结尾加句点

若用户需要定制更多内容，可以学习 `bst` 文件的语法并修改<sup>[4-6]</sup>，或者联系作者。

## 8 相关工作

TeX 社区也有其他关于 GB/T 7714 系列参考文献标准的工作。2005 年吴凯<sup>[7]</sup>发布了基于 GB/T 7714—2005 的 BibTeX 样式，支持顺序编码制和著者出版年制两种风格。李志奇<sup>[8]</sup>发布了严格遵循 GB/T 7714—2005 的 BibLaTeX 的样式。胡海星<sup>[9]</sup>提供了另一个 BibTeX 实现，还给每行 bst 代码写了 java 语言注释。沈周<sup>[10]</sup>基于 biblatex-casperiector<sup>[11]</sup> 进行修改，以符合国标的格式。胡振震发布了符合 GB/T 7714—2015 标准的 BibLaTeX 参考文献样式<sup>[12]</sup>，并进行了比较完善的持续维护。

## 参考文献

- [1] 中国标准化委员会. 信息与文献 参考文献著录规则: GB/T 7714—2015[S]. 北京: 中国标准出版社, 2015.
- [2] PATASHNIK O. BibTeXing[M/OL]. 1988. <http://mirrors.ctan.org/biblio/bibtex/base/btxdoc.pdf>.
- [3] DALY P W. Natural sciences citations and references[M/OL]. 1999. <http://mirrors.ctan.org/macros/latex/contrib/natbib/natbib.pdf>.
- [4] PATASHNIK O. Designing BibTeX styles[M/OL]. 1988. <http://mirrors.ctan.org/biblio/bibtex/base/btxhak.pdf>.
- [5] MARKEY N. Tame the beast[M/OL]. 2003. [http://mirrors.ctan.org/info/bibtex/tamethebeast/ttb\\_en.pdf](http://mirrors.ctan.org/info/bibtex/tamethebeast/ttb_en.pdf).
- [6] MITTELBAACH F, GOOSSENS M, BRAAMS J, et al. The L<sup>A</sup>T<sub>E</sub>X companion[M]. 2nd ed. Reading, MA, USA: Addison-Wesley, 2004.
- [7] 吴凯. 发布 GBT7714-2005.bst version1 Beta 版 [EB/OL]. 2006. CTeX 论坛 (已关闭) .
- [8] 李志奇. 基于 biblatex 的符合 GBT7714—2005 的中文文献生成工具 [EB/OL]. 2013. CTeX 论坛 (已关闭) .
- [9] 胡海星. A GB/T 7714—2005 national standard compliant BibTeX style[EB/OL]. 2013. <https://github.com/Haixing-Hu/GBT7714-2005-BibTeX-Style>.
- [10] 沈周. 基于 casperiector 改写的符合 GB/T 7714—2005 标准的参考文献格式 [EB/OL]. 2016. <https://github.com/szsdk/biblatex-gbt77142005>.

- [11] VECTOR C T. biblatex 参考文献和引用样式: caspervector[M/OL]. 2012. <http://mirrors.ctan.org/macros/latex/contrib/biblatex-contrib/biblatex-caspervector/doc/caspervector.pdf>.
- [12] 胡振震. 符合 GB/T 7714—2015 标准的 biblatex 参考文献样式 [M/OL]. 2016. <http://mirrors.ctan.org/macros/latex/contrib/biblatex-contrib/biblatex-gb7714-2015/biblatex-gb7714-2015.pdf>.

## A 宏包的代码实现

兼容过时的接口

```
1 (*package)
2 \newif\ifgbt@legacy@interface
3 \newif\ifgbt@mmxv
4 \newif\ifgbt@numerical
5 \newif\ifgbt@super
6 \newcommand\gbt@obsolete@option[1]{%
7   \PackageWarning{gbt7714}{The option "#1" is obsolete}%
8 }
9 \DeclareOption{2015}{%
10   \gbt@obsolete@option{2015}%
11   \gbt@legacy@interfacetrue
12   \gbt@mmxvtrue
13 }
14 \DeclareOption{2005}{%
15   \gbt@obsolete@option{2005}%
16   \gbt@legacy@interfacetrue
17   \gbt@mmxvfalse
18 }
19 \DeclareOption{super}{%
20   \gbt@obsolete@option{super}%
21   \gbt@legacy@interfacetrue
22   \gbt@numericaltrue
23   \gbt@supertrue
24 }
25 \DeclareOption{numbers}{%
26   \gbt@obsolete@option{numbers}%
27   \gbt@legacy@interfacetrue
28   \gbt@numericaltrue
29   \gbt@superfalse
30 }
31 \DeclareOption{authoryear}{%
32   \gbt@obsolete@option{authoryear}%
33   \gbt@legacy@interfacetrue
34   \gbt@numericalfalse
35 }
```

将选项传递给 natbib

```
36 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{natbib}}
37 \ProcessOptions\relax
```



调用宏包，注意只需要 `compress` 不需要 `sort`。

```
38 \RequirePackage{natbib}
```

```
39 \RequirePackage{url}
```

如果将 `compress` 传给 `natbib` 容易导致 `option clash`。这里直接修改内部命令。

```
40 \def\NAT@cmprs{\@ne}
```

`\citestyle` 定义接口切换引用文献的标注法,可用 `\citestyle` 调用 `numerical` 或 `authoryear`, 参见 `natbib`。

```
41 \renewcommand\newblock{\space}
```

```
42 \newcommand\bibstyle@super{\bibpunct{[ ]}{,}{s}{,}{\textsuperscript{,}}}
```

```
43 \newcommand\bibstyle@numbers{\bibpunct{[ ]}{,}{n}{,}{,}{,}}
```

```
44 \newcommand\bibstyle@authoryear{\bibpunct{({})}{;}{a}{,}{,}{,}}
```

```
45 \newcommand\bibstyle@inline{\bibstyle@numbers}
```

在使用 `\bibliographystyle` 时自动切换引用文献的标注的样式。

```
46 \@namedef{bibstyle@gbt7714-numerical}{\bibstyle@super}
```

```
47 \@namedef{bibstyle@gbt7714-author-year}{\bibstyle@authoryear}
```

```
48 \@namedef{bibstyle@gbt7714-2005-numerical}{\bibstyle@super}
```

```
49 \@namedef{bibstyle@gbt7714-2005-author-year}{\bibstyle@authoryear}
```

`\cite` 下面修改 `natbib` 的引用格式。为了减少依赖的宏包，这里直接重定义命令不使用 `\patchcmd`。

`Super` 样式的 `\citep` 的页码也为上标。另外加上 `\kern\p@` 去掉上标式引用后与中文之间多余的空格，参考 [tuna/thuthesis#624](#)。

```
50 \renewcommand\NAT@citesuper[3]{%
```

```
51 \ifNAT@swa
```

```
52 \if*#2*\else
```

```
53 #2\NAT@spacechar
```

```
54 \fi
```

```
55 % \unskip\kern\p@\textsuperscript{\NAT@open#1\NAT@close}%
```

```
56 % \if*#3*\else\NAT@spacechar#3\fi\else #1\fi\endgroup}
```

```
57 \unskip\kern\p@
```

```
58 \textsuperscript{%
```

```
59 \NAT@open
```

```
60 #1%
```

```
61 \NAT@close
```

```
62 \if*#3*\else
```

```
63 #3%
```

```
64 \fi
```

```
65 }%
```

```

66 \kern\p@
67 \else
68 #1%
69 \fi
70 \endgroup
71 }

```

将 numbers 样式的 \citep 的页码置于括号外。

```

72 \renewcommand\NAT@citenum[3]{%
73 \ifNAT@swa
74 \NAT@@open
75 \if*#2*\else
76 #2\NAT@spacechar
77 \fi
78 % #1\if*#3*\else\NAT@cmt#3\fi\NAT@close\else#1\fi\endgroup}
79 #1\NAT@close
80 \if*#3*\else
81 \textsuperscript{#3}%
82 \fi
83 \else
84 #1%
85 \fi
86 \endgroup
87 }

```

Numerical 模式的 \citet 的页码：

```

88 \def\NAT@citexnum[#1][#2]#3{%
89 \NAT@reset@parser
90 \NAT@sort@cites{#3}%
91 \NAT@reset@citea
92 \@cite{\def\NAT@num{-1}\let\NAT@last@yr\relax\let\NAT@nm\@empty
93 \@for\@citeb:=\NAT@cite@list\do
94 {\@safe@activetrue
95 \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
96 \@safe@activesfalse
97 \@ifundefined{b@\@citeb\@extra@b@citeb}{%
98 {\reset@font\bfseries?}
99 \NAT@citeundefined\PackageWarning{natbib}%
100 {Citation '\@citeb' on page \thepage \space undefined}}%
101 {\let\NAT@last@num\NAT@num\let\NAT@last@nm\NAT@nm
102 \NAT@parse{\@citeb}%
103 \ifNAT@longnames\@ifundefined{bv@\@citeb\@extra@b@citeb}{%
104 \let\NAT@name=\NAT@all@names

```

```

105     \global\@namedef{bv@\@citeb\@extra@b@citeb}{}}{}%
106     \fi
107     \ifNAT@full\let\NAT@nm\NAT@all@names\else
108         \let\NAT@nm\NAT@name\fi
109     \ifNAT@swa
110         \@ifnum{\NAT@ctype>\@ne}{%
111             \citea
112             \NAT@hyper@{\@ifnum{\NAT@ctype=\tw@}{\NAT@test{\NAT@ctype}}{\NAT@alias}}%
113         }{%
114             \@ifnum{\NAT@cmprs>\z@}{%
115                 \NAT@ifcat@num\NAT@num
116                 {\let\NAT@nm=\NAT@num}%
117                 {\def\NAT@nm{-2}}%
118                 \NAT@ifcat@num\NAT@last@num
119                 {\@tempcnta=\NAT@last@num\relax}%
120                 {\@tempcnta\m@ne}%
121                 \@ifnum{\NAT@nm=\@tempcnta}{%
122                     \@ifnum{\NAT@merge>\@ne}{\NAT@last@yr@mbbox}%
123                 }{%
124                     \advance\@tempcnta by\@ne
125                     \@ifnum{\NAT@nm=\@tempcnta}{%

```

在顺序编码制下，**natbib** 只有在三个以上连续文献引用才会使用连接号，这里修改为允许两个引用使用连接号。

```

126         % \ifx\NAT@last@yr\relax
127         %     \def@NAT@last@yr{\@citea}%
128         % \else
129         %     \def@NAT@last@yr{--\NAT@penalty}%
130         % \fi
131         \def@NAT@last@yr{-\NAT@penalty}%
132     }{%
133         \NAT@last@yr@mbbox
134     }%
135 }%
136 }{%
137     \@tempswatrue
138     \@ifnum{\NAT@merge>\@ne}{\@ifnum{\NAT@last@num=\NAT@num\relax}{\@tempswafalse}{}}{}%
139     \if@tempswa\NAT@citea@mbbox\fi
140 }%
141 }%
142 \NAT@def@citea
143 \else
144     \ifcase\NAT@ctype

```

```

145         \ifx\NAT@last@nm\NAT@nm \NAT@yrsep\NAT@penalty\NAT@space\else
146         \@citea \NAT@test{\@ne}\NAT@spacechar\NAT@mbox{\NAT@super@kern\NAT@@open}%
147         \fi
148         \if*#1*\else#1\NAT@spacechar\fi
149         \NAT@mbox{\NAT@hyper@{\@citenumfont{\NAT@num}}}%
150         \NAT@def@citea@box
151     \or
152         \NAT@hyper@citea@space{\NAT@test{\NAT@ctype}}%
153     \or
154         \NAT@hyper@citea@space{\NAT@test{\NAT@ctype}}%
155     \or
156         \NAT@hyper@citea@space\NAT@alias
157     \fi
158 \fi
159 }%
160 }%
161 \@ifnum{\NAT@cmprs>\z@}{\NAT@last@yr}{}%
162 \ifNAT@swa\else

```

将页码放在括号外边，并且置于上标。

```

163 % \@ifnum{\NAT@ctype=\z@}{%
164 %     \if*#2*\else\NAT@cmt#2\fi
165 % }{%
166 \NAT@mbox{\NAT@close}%
167 \@ifnum{\NAT@ctype=\z@}{%
168     \if*#2*\else
169         \textsuperscript{#2}%
170     \fi
171 }{%
172 \NAT@super@kern
173 \fi
174 }{#1}{#2}%
175 }%

```

Author-year 模式的 \citep 的页码：

```

176 \renewcommand\NAT@cite%
177 [3]{\ifNAT@swa\NAT@@open\if*#2*\else#2\NAT@spacechar\fi
178     #1\NAT@close\if*#3*\else\textsuperscript{#3}\fi\else#1\fi\endgroup}

```

Author-year 模式的 \citet 的页码：

```

179 \def\NAT@citex%
180 [#1][#2]#3{%
181 \NAT@reset@parser

```

```

182 \NAT@sort@cites{#3}%
183 \NAT@reset@citea
184 \@cite{\let\NAT@nm\@empty\let\NAT@year\@empty
185 \@for\@citeb:=\NAT@cite@list\do
186 {\@safe@activestrue
187 \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
188 \@safe@activesfalse
189 \@ifundefined{b@\@citeb\@extra@b@citeb}{\@citea%
190 {\reset@font\bfseries ?}\NAT@citeundefined
191 \PackageWarning{natbib}%
192 {Citation '\@citeb' on page \thepage \space undefined}}\def\NAT@date{}}%
193 {\let\NAT@last@nm=\NAT@nm\let\NAT@last@yr=\NAT@year
194 \NAT@parse{\@citeb}%
195 \ifNAT@longnames\@ifundefined{bv@\@citeb\@extra@b@citeb}{%
196 \let\NAT@name=\NAT@all@names
197 \global\@namedef{bv@\@citeb\@extra@b@citeb}{}}{}%
198 \fi
199 \ifNAT@full\let\NAT@nm\NAT@all@names\else
200 \let\NAT@nm\NAT@name\fi
201 \ifNAT@swa\ifcase\NAT@ctype
202 \if\relax\NAT@date\relax
203 \@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}\NAT@date}%
204 \else
205 \ifx\NAT@last@nm\NAT@nm\NAT@yrsep
206 \ifx\NAT@last@yr\NAT@year
207 \def\NAT@temp{}}{}%
208 \ifx\NAT@temp\NAT@exlab\PackageWarningNoLine{natbib}%
209 {Multiple citation on page \thepage: same authors and
210 year\MessageBreak without distinguishing extra
211 letter,\MessageBreak appears as question mark}\fi
212 \NAT@hyper@{\NAT@exlab}%
213 \else\unskip\NAT@spacechar
214 \NAT@hyper@{\NAT@date}%
215 \fi
216 \else
217 \@citea\NAT@hyper@{%
218 \NAT@nmfmt{\NAT@nm}%
219 \hyper@natlinkbreak{%
220 \NAT@aysep\NAT@spacechar}{\@citeb\@extra@b@citeb
221 }%
222 \NAT@date
223 }%

```

```

224     \fi
225   \fi
226   \or\@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
227   \or\@citea\NAT@hyper@{\NAT@date}%
228   \or\@citea\NAT@hyper@{\NAT@alias}%
229   \fi \NAT@def@citea
230   \else
231     \ifcase\NAT@ctype
232     \if\relax\NAT@date\relax
233       \@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
234     \else
235       \ifx\NAT@last@nm\NAT@nm\NAT@yrsep
236         \ifx\NAT@last@yr\NAT@year
237           \def\NAT@temp{{?}}%
238           \ifx\NAT@temp\NAT@exlab\PackageWarningNoLine{natbib}%
239             {Multiple citation on page \thepage: same authors and
240             year\MessageBreak without distinguishing extra
241             letter,\MessageBreak appears as question mark}\fi
242           \NAT@hyper@{\NAT@exlab}%
243         \else
244           \unskip\NAT@spacechar
245           \NAT@hyper@{\NAT@date}%
246         \fi
247       \else
248         \@citea\NAT@hyper@{%
249         \NAT@nmfmt{\NAT@nm}%
250         \hyper@natlinkbreak{\NAT@spacechar\NAT@@open\if*#1*\else#1\NAT@spacechar\fi}%
251         {\@citeb\@extra@b@citeb}%
252         \NAT@date
253         }%
254       \fi
255     \fi
256     \or\@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
257     \or\@citea\NAT@hyper@{\NAT@date}%
258     \or\@citea\NAT@hyper@{\NAT@alias}%
259     \fi
260     \if\relax\NAT@date\relax
261       \NAT@def@citea
262     \else
263       \NAT@def@citea@close
264     \fi
265   \fi

```

```
266 }}\ifNAT@swa\else
```

将页码放在括号外边，并且置于上标。

```
267 % \if*#2*\else\NAT@cmt#2\fi
268 \if\relax\NAT@date\relax\else\NAT@@close\fi
269 \if*#2*\else\textsuperscript{#2}\fi
270 \fi}{#1}{#2}}
```

thebibliography 参考文献列表的标签左对齐

```
271 \renewcommand\@biblabel[1]{#1}\hfill}
```

\url 使用 xurl 宏包的方法，增加 URL 可断行的位置。

```
272 \g@addto@macro\UrlBreaks{%
273 \do0\do1\do2\do3\do4\do5\do6\do7\do8\do9%
274 \doA\doB\doC\doD\doE\doF\doG\doH\doI\doJ\doK\doL\doM
275 \doN\doO\doP\doQ\doR\doS\doT\doU\doV\doW\doX\doY\doZ
276 \doa\dob\doc\do d\doe\do f\dog\do h\do i\do j\do k\do l\do m
277 \do n\do o\do p\do q\do r\do s\do t\do u\do v\do w\do x\do y\do z
278 }
279 \Urlmuskip=0mu plus 0.1mu
```

兼容 v2.0 前过时的接口：

```
280 \newif\ifgibt@bib@style@written
281 \@ifpackageloaded{chapterbib}{}%
282 \def\bibliography#1{%
283 \ifgibt@bib@style@written\else
284 \bibliographystyle{gbt7714-numerical}%
285 \fi
286 \if@filesw
287 \immediate\write\@auxout{\string\bibdata{\zap@space#1 \@empty}}%
288 \fi
289 \@input@{\jobname.bbl}}
290 \def\bibliographystyle#1{%
291 \gibt@bib@style@writtenttrue
292 \ifx\@begindocumenthook\@undefined\else
293 \expandafter\AtBeginDocument
294 \fi
295 {\if@filesw
296 \immediate\write\@auxout{\string\bibstyle{#1}}%
297 \fi}%
298 }%
299 }
300 \ifgibt@legacy@interface
```

```

301 \ifgbt@numerical
302   \ifgbt@super\else
303     \citestyle{numbers}
304   \fi
305   \bibliographystyle{gbt7714-numerical}
306 \else
307   \bibliographystyle{gbt7714-author-year}
308 \fi
309 \fi
310 </package>

```

## B BibTeX 样式的代码实现

### B.1 自定义选项

bst 这里定义了一些变量用于定制样式，可以在下面的 `load.config` 函数中选择是否启用。

```

311 (*author-year | numerical)
312 INTEGERS {
313   citation.et.al.min
314   citation.et.al.use.first
315   bibliography.et.al.min
316   bibliography.et.al.use.first
317   uppercase.name
318   terms.in.macro
319   year.after.author
320   period.after.author
321   italic.book.title
322   sentence.case.title
323   link.title
324   title.in.journal
325   show.mark
326   space.before.mark
327   show.medium.type
328   slash.for.extraction
329   in.booktitle
330   short.journal
331   italic.journal
332   bold.journal.volume
333   show.missing.address.publisher
334   space.before.pages
335   only.start.page
336   wave.dash.in.pages
337   show.urldate
338   show.url
339   show.doi
340   show.preprint
341   show.note
342   show.english.translation

```



```

343 end.with.period
344 (*author-year)
345 lang.zh.order
346 lang.ja.order
347 lang.en.order
348 lang.ru.order
349 lang.other.order
350 </author-year>
351 }
352

```

下面每个变量若被设为 **#1** 则启用该项，若被设为 **#0** 则不启用。默认的值是严格遵循国标的配置。

```

353 FUNCTION {load.config}
354 {

```

如果姓名的数量大于等于 **et.al.min**，只著录前 **et.al.use.first** 个，其后加“et al.”或“等”。

```

355 (*!ucas)
356 #2 'citation.et.al.min :=
357 #1 'citation.et.al.use.first :=
358 </!ucas)
359 (*ucas)
360 #3 'citation.et.al.min :=
361 #1 'citation.et.al.use.first :=
362 </ucas)
363 #4 'bibliography.et.al.min :=
364 #3 'bibliography.et.al.use.first :=

```

英文姓名转为全大写：

```

365 (*!(no-uppercase | thu))
366 #1 'uppercase.name :=
367 </!(no-uppercase | thu))
368 (*no-uppercase | thu)
369 #0 'uppercase.name :=
370 </no-uppercase | thu)

```

使用 TeX 宏输出“和”、“等”

```

371 (*!(macro | ucas))
372 #0 'terms.in.macro :=
373 </!(macro | ucas))
374 (*macro | ucas)
375 #1 'terms.in.macro :=
376 </macro | ucas)

```

将年份置于著者后面（著者-出版年制默认）

```

377 (*numerical | ucas)
378 #0 'year.after.author :=
379 </numerical | ucas)
380 (*author-year&!ucas)
381 #1 'year.after.author :=
382 </author-year&!ucas)

```

采用著者-出版年制时，作者姓名与年份之间使用句点连接：

```

383 < *numerical >
384 #1 'period.after.author :=
385 < /numerical >
386 < *author-year >
387 < *2015&!(period | ustc) >
388 #0 'period.after.author :=
389 < /2015&!(period | ustc) >
390 < *period | 2005 | ustc >
391 #1 'period.after.author :=
392 < /period | 2005 | ustc >
393 < /author-year >

```

书名使用斜体:

```

394 < *!italic-book-title >
395 #0 'italic.book.title :=
396 < /!italic-book-title >
397 < *italic-book-title >
398 #1 'italic.book.title :=
399 < /italic-book-title >

```

英文标题转为 sentence case (句首字母大写, 其余小写):

```

400 < *!no-sentence-case >
401 #1 'sentence.case.title :=
402 < /!no-sentence-case >
403 < *no-sentence-case >
404 #0 'sentence.case.title :=
405 < /no-sentence-case >

```

在标题添加超链接:

```

406 < *!link-title >
407 #0 'link.title :=
408 < /!link-title >
409 < *link-title >
410 #1 'link.title :=
411 < /link-title >

```

期刊是否含标题:

```

412 < *!no-title-in-journal >
413 #1 'title.in.journal :=
414 < /!no-title-in-journal >
415 < *no-title-in-journal >
416 #0 'title.in.journal :=
417 < /no-title-in-journal >

```

著录文献类型标识 (比如“[M/OL]”):

```

418 < *!no-mark >
419 #1 'show.mark :=
420 < /!no-mark >
421 < *no-mark >
422 #0 'show.mark :=
423 < /no-mark >

```

文献类型标识前是否有空格:

```

424 < *!space-before-mark >
425 #0 'space.before.mark :=

```

```

426 <!/space-before-mark>
427 <*space-before-mark>
428 #1 'space.before.mark :=
429 </space-before-mark>

```

是否显示载体类型标识（比如“/OL”）：

```

430 <*!no-medium-type>
431 #1 'show.medium.type :=
432 </!no-medium-type>
433 <*no-medium-type>
434 #0 'show.medium.type :=
435 </no-medium-type>

```

使用“//”表示析出文献

```

436 <*!no-slash>
437 #1 'slash.for.extraction :=
438 </!no-slash>
439 <*no-slash>
440 #0 'slash.for.extraction :=
441 </no-slash>

```

使用“In:”表示析出文献

```

442 #0 'in.booktitle :=

```

期刊名使用缩写：

```

443 <*!short-journal>
444 #0 'short.journal :=
445 </!short-journal>
446 <*short-journal>
447 #1 'short.journal :=
448 </short-journal>

```

期刊名使用斜体：

```

449 <*!italic-journal>
450 #0 'italic.journal :=
451 </!italic-journal>
452 <*italic-journal>
453 #1 'italic.journal :=
454 </italic-journal>

```

期刊的卷使用粗体：

```

455 #0 'bold.journal.volume :=

```

无出版地或出版者时，著录“出版地不详”，“出版者不详”，“S.l.”或“s.n.”：

```

456 <*!sl-sn>
457 #0 'show.missing.address.publisher :=
458 </!sl-sn>
459 <*sl-sn>
460 #1 'show.missing.address.publisher :=
461 </sl-sn>

```

页码与前面的冒号之间是否有空格：

```

462 <*!no-space-before-pages>
463 #1 'space.before.pages :=
464 </!no-space-before-pages>

```

```

465 <!*no-space-before-pages>
466 #0 'space.before.pages :=
467 </no-space-before-pages>

```

页码是否只含起始页：

```

468 <!*only-start-page>
469 #0 'only.start.page :=
470 </!only-start-page>
471 <*only-start-page>
472 #1 'only.start.page :=
473 </only-start-page>

```

起止页码使用波浪号：

```

474 <!*wave-dash-in-pages>
475 #0 'wave.dash.in.pages :=
476 </!wave-dash-in-pages>
477 <*wave-dash-in-pages>
478 #1 'wave.dash.in.pages :=
479 </wave-dash-in-pages>

```

是否著录非电子文献的引用日期：

```

480 <!*no-urldate>
481 #1 'show.urldate :=
482 </!no-urldate>
483 <*no-urldate>
484 #0 'show.urldate :=
485 </no-urldate>

```

是否著录 URL：

```

486 <!*no-url>
487 #1 'show.url :=
488 </!no-url>
489 <*no-url>
490 #0 'show.url :=
491 </no-url>

```

是否著录 DOI：

```

492 <*!(no-doi | 2005)>
493 #1 'show.doi :=
494 </!(no-doi | 2005)>
495 <*no-doi | 2005>
496 #0 'show.doi :=
497 </no-doi | 2005>

```

是否著录 e-print：

```

498 <!*preprint>
499 #0 'show.preprint :=
500 </!preprint>
501 <*preprint>
502 #1 'show.preprint :=
503 </preprint>

```

在每一条文献最后输出注释（note）的内容：

```

504 #0 'show.note :=

```

中文文献是否显示英文翻译

```
505 \<!*show-english-translation>
506 #0 'show.english.translation :=
507 \</!show-english-translation>
508 \<!*show-english-translation>
509 #1 'show.english.translation :=
510 \</show-english-translation>
```

结尾加句点

```
511 \<!*no-period-at-end>
512 #1 'end.with.period :=
513 \</!no-period-at-end>
514 \<!*no-period-at-end>
515 #0 'end.with.period :=
516 \</no-period-at-end>
```

参考文献表按照“著者-出版年”组织时，各个文种的顺序：

```
517 \<!*author-year>
518 #1 'lang.zh.order :=
519 #2 'lang.ja.order :=
520 #3 'lang.en.order :=
521 #4 'lang.ru.order :=
522 #5 'lang.other.order :=
523 \</author-year>
524 }
525
```

## B.2 The ENTRY declaration

Like Scribe's (according to pages 231-2 of the April '84 edition), but no full-author or editors fields because BibTeX does name handling. The `annote` field is commented out here because this family doesn't include an annotated bibliography style. And in addition to the fields listed here, BibTeX has a built-in `crossref` field, explained later.

```
526 ENTRY
527 { address
528   archivePrefix
529   author
530   booktitle
531   date
532   doi
533   edition
534   editor
535   eprint
536   eprinttype
537   howpublished
538   institution
539   journal
540   journaltitle
541   key
542   langid
```

```

543     language
544     location
545     mark
546     medium
547     note
548     number
549     organization
550     pages
551     publisher
552     school
553     series
554     shortjournal
555     title
556     translation
557     translator
558     url
559     urldate
560     volume
561     year
562 }
563 { entry.lang entry.is.electronic is.pure.electronic entry.numbered }

```

These string entry variables are used to form the citation label. In a storage pinch, sort.label can be easily computed on the fly.

```

564 { label extra.label sort.label short.label short.list entry.mark entry.url }
565

```

### B.3 Entry functions

Each entry function starts by calling output.bibitem, to write the \bibitem and its arguments to the .BBL file. Then the various fields are formatted and printed by output or output.check. Those functions handle the writing of separators (commas, periods, \newblock's), taking care not to do so when they are passed a null string. Finally, fin.entry is called to add the final period and finish the entry.

A bibliographic reference is formatted into a number of 'blocks': in the open format, a block begins on a new line and subsequent lines of the block are indented. A block may contain more than one sentence (well, not a grammatical sentence, but something to be ended with a sentence ending period). The entry functions should call new.block whenever a block other than the first is about to be started. They should call new.sentence whenever a new sentence is to be started. The output functions will ensure that if two new.sentence's occur without any non-null string being output between them then there won't be two periods output. Similarly for two successive new.block's.

The output routines don't write their argument immediately. Instead, by convention, that argument is saved on the stack to be output next time (when we'll know

what separator needs to come after it). Meanwhile, the output routine has to pop the pending output off the stack, append any needed separator, and write it.

To tell which separator is needed, we maintain an `output.state`. It will be one of these values: `before.all` just after the `\bibitem` `mid.sentence` in the middle of a sentence: comma needed if more sentence is output `after.sentence` just after a sentence: period needed `after.block` just after a block (and sentence): period and `\newblock` needed. Note: These styles don't use `after.sentence`

VAR: `output.state` : INTEGER – state variable for output

The `output.nonnull` function saves its argument (assumed to be nonnull) on the stack, and writes the old saved value followed by any needed separator. The ordering of the tests is decreasing frequency of occurrence.

由于专著中的析出文献需要用到很特殊的“//”，所以我又加了一个 `after.slash`。其他需要在特定符号后面输出，所以写了一个 `output.after`。

```
output.nonnull(s) ==
BEGIN
  s := argument on stack
  if output.state = mid.sentence then
    write$(pop() * ", ")
    -- "pop" isn't a function: just use stack top
  else
    if output.state = after.block then
      write$(add.period$(pop()))
      newline$
      write$("\newblock ")
    else
      if output.state = before.all then
        write$(pop())
      else
        -- output.state should be after.sentence
        write$(add.period$(pop()) * " ")
      fi
    fi
    output.state := mid.sentence
  fi
  push s on stack
END
```

The output function calls `output.nonnull` if its argument is non-empty; its argument may be a missing field (thus, not necessarily a string)

```
output(s) ==
BEGIN
  if not empty$(s) then output.nonnull(s)
  fi
END
```

The `output.check` function is the same as the `output` function except that, if necessary, `output.check` warns the user that the `t` field shouldn't be empty (this is because

it probably won't be a good reference without the field; the entry functions try to make the formatting look reasonable even when such fields are empty).

```
output.check(s,t) ==
BEGIN
  if empty$(s) then
    warning$("empty " * t * " in " * cite$)
  else output.nonnull(s)
  fi
END
```

The output.bibitem function writes the \bibitem for the current entry (the label should already have been set up), and sets up the separator state for the output functions. And, it leaves a string on the stack as per the output convention.

```
output.bibitem ==
BEGIN
  newline$
  write$("\bibitem[")      % for alphabetic labels,
  write$(label)           % these three lines
  write$("{")             % are used
  write$("\bibitem{")      % this line for numeric labels
  write$(cite$)
  write$("}")
  push "" on stack
  output.state := before.all
END
```

The fin.entry function finishes off an entry by adding a period to the string remaining on the stack. If the state is still before.all then nothing was produced for this entry, so the result will look bad, but the user deserves it. (We don't omit the whole entry because the entry was cited, and a bibitem is needed to define the citation label.)

```
fin.entry ==
BEGIN
  write$(add.period$(pop()))
  newline$
END
```

The new.block function prepares for a new block to be output, and new.sentence prepares for a new sentence.

```
new.block ==
BEGIN
  if output.state <> before.all then
    output.state := after.block
  fi
END
```

```
new.sentence ==
BEGIN
```



```

        if output.state <> after.block then
            if output.state <> before.all then
                output.state := after.sentence
            fi
        fi
    fi
END

```

```

566 INTEGERS { output.state before.all mid.sentence after.sentence after.block after.slash }
567
568 INTEGERS { lang.zh lang.ja lang.en lang.ru lang.other }
569
570 INTEGERS { charptr len }
571
572 FUNCTION {init.state.consts}
573 { #0 'before.all :=
574   #1 'mid.sentence :=
575   #2 'after.sentence :=
576   #3 'after.block :=
577   #4 'after.slash :=
578   #3 'lang.zh :=
579   #4 'lang.ja :=
580   #1 'lang.en :=
581   #2 'lang.ru :=
582   #0 'lang.other :=
583 }
584

```

下面是一些常量的定义

```

585 FUNCTION {bbl.anonymous}
586 { entry.lang lang.zh =
587   { " 佚名" }
588   { "Anon" }
589   if$
590 }
591
592 FUNCTION {bbl.space}
593 { entry.lang lang.zh =
594   { "\ " }
595   { " " }
596   if$
597 }
598
599 FUNCTION {bbl.and}
600 { "" }
601
602 FUNCTION {bbl.et.al}
603 { entry.lang lang.zh =
604   { " 等" }
605   { entry.lang lang.ja =
606     { " 他" }
607     { entry.lang lang.ru =
608       { "и др" }
609       { "et~al." }
610       if$
611     }

```

```

612     if$
613   }
614   if$
615 }
616
617 FUNCTION {citation.and}
618 { terms.in.macro
619   { "{\biband}" }
620   'bbl.and
621   if$
622 }
623
624 FUNCTION {citation.et.al}
625 { terms.in.macro
626   { "{\bibetal}" }
627   'bbl.et.al
628   if$
629 }
630
631 FUNCTION {bbl.colon} { ": " }
632
633 FUNCTION {bbl.pages.colon}
634 { space.before.pages
635   { ": " }
636   { ":\allowbreak " }
637   if$
638 }
639
640 (*!2005)
641 FUNCTION {bbl.wide.space} { "\quad " }
642 </!2005>
643 (*2005)
644 FUNCTION {bbl.wide.space} { "\ " }
645 </2005>
646
647 FUNCTION {bbl.slash} { "//\allowbreak " }
648
649 FUNCTION {bbl.sine.loco}
650 { entry.lang lang.zh =
651   { "[出版地不详]" }
652   { "[S.l.]" }
653   if$
654 }
655
656 FUNCTION {bbl.sine.nomine}
657 { entry.lang lang.zh =
658   { "[出版者不详]" }
659   { "[s.n.]" }
660   if$
661 }
662
663 FUNCTION {bbl.sine.loco.sine.nomine}
664 { entry.lang lang.zh =
665   { "[出版地不详: 出版者不详]" }
666   { "[S.l.: s.n.]" }

```

```

667  if$
668 }
669

```

These three functions pop one or two (integer) arguments from the stack and push a single one, either 0 or 1. The 'skip\$ in the 'and' and 'or' functions are used because the corresponding if\$ would be idempotent

```

670 FUNCTION {not}
671 {   { #0 }
672   { #1 }
673   if$
674 }
675
676 FUNCTION {and}
677 {   'skip$
678   { pop$ #0 }
679   if$
680 }
681
682 FUNCTION {or}
683 {   { pop$ #1 }
684   'skip$
685   if$
686 }
687
688 STRINGS { x y }
689
690 FUNCTION {contains}
691 { 'y :=
692   'x :=
693   y text.length$ 'len :=
694   x text.length$ len - #1 + 'charptr :=
695   { charptr #0 >
696     x charptr len substring$ y = not
697     and
698   }
699   { charptr #1 - 'charptr := }
700   while$
701   charptr #0 >
702 }
703

```

the variables s and t are temporary string holders

```

704 STRINGS { s t }
705
706 FUNCTION {output.nonnull}
707 { 's :=
708   output.state mid.sentence =
709   { ", " * write$ }
710   { output.state after.block =
711     { add.period$ write$
712       newline$
713       "\newblock " write$
714     }

```

```

715         { output.state before.all =
716             'write$
717             { output.state after.slash =
718                 { bbl.slash * write$
719                     newline$
720                 }
721                 { add.period$ " " * write$ }
722             if$
723         }
724     if$
725 }
726 if$
727     mid.sentence 'output.state :=
728 }
729 if$
730 s
731 }
732
733 FUNCTION {output}
734 { duplicate$ empty$
735     'pop$
736     'output.nonnull
737     if$
738 }
739
740 FUNCTION {output.after}
741 { 't :=
742     duplicate$ empty$
743     'pop$
744     { 's :=
745         output.state mid.sentence =
746         { t * write$ }
747         { output.state after.block =
748             { add.period$ write$
749                 newline$
750                 "\newblock " write$
751             }
752             { output.state before.all =
753                 'write$
754                 { output.state after.slash =
755                     { bbl.slash * write$ }
756                     { add.period$ " " * write$ }
757                 if$
758             }
759             if$
760         }
761         if$
762         mid.sentence 'output.state :=
763     }
764     if$
765     s
766 }
767 if$
768 }
769

```

```

770 FUNCTION {output.check}
771 { 't :=
772   duplicate$ empty$
773   { pop$ "empty " t * " in " * cite$ * warning$ }
774   'output.nonnull
775   if$
776 }
777

```

This function finishes all entries.

```

778 FUNCTION {fin.entry}
779 { end.with.period
780   'add.period$
781   'skip$
782   if$
783   write$
784   show.english.translation entry.lang lang.zh = and
785   { ""
786     write$
787   }
788   'skip$
789   if$
790   newline$
791 }
792
793 FUNCTION {new.block}
794 { output.state before.all =
795   'skip$
796   { output.state after.slash =
797     'skip$
798     { after.block 'output.state := }
799     if$
800   }
801   if$
802 }
803
804 FUNCTION {new.sentence}
805 { output.state after.block =
806   'skip$
807   { output.state before.all =
808     'skip$
809     { output.state after.slash =
810       'skip$
811       { after.sentence 'output.state := }
812       if$
813     }
814     if$
815   }
816   if$
817 }
818
819 FUNCTION {new.slash}
820 { output.state before.all =
821   'skip$
822   { slash.for.extraction

```

```

823         { after.slash 'output.state := }
824         { after.block 'output.state := }
825     if$
826 }
827 if$
828 }
829

```

Sometimes we begin a new block only if the block will be big enough. The `new.block.checka` function issues a `new.block` if its argument is nonempty; `new.block.checkb` does the same if either of its TWO arguments is nonempty.

```

830 FUNCTION {new.block.checka}
831 { empty$
832   'skip$
833   'new.block
834   if$
835 }
836
837 FUNCTION {new.block.checkb}
838 { empty$
839   swap$ empty$
840   and
841   'skip$
842   'new.block
843   if$
844 }
845

```

The `new.sentence.check` functions are analogous.

```

846 FUNCTION {new.sentence.checka}
847 { empty$
848   'skip$
849   'new.sentence
850   if$
851 }
852
853 FUNCTION {new.sentence.checkb}
854 { empty$
855   swap$ empty$
856   and
857   'skip$
858   'new.sentence
859   if$
860 }
861

```

## B.4 Formatting chunks

Here are some functions for formatting chunks of an entry. By convention they either produce a string that can be followed by a comma or period (using `add.period$`, so it is OK to end in a period), or they produce the null string.

A useful utility is the `field.or.null` function, which checks if the argument is the result of pushing a ‘missing’ field (one for which no assignment was made when the current entry was read in from the database) or the result of pushing a string having no non-white-space characters. It returns the null string if so, otherwise it returns the field string. Its main (but not only) purpose is to guarantee that what’s left on the stack is a string rather than a missing field.

```
field.or.null(s) ==
BEGIN
    if empty$(s) then return ""
    else return s
END
```

Another helper function is `emphasize`, which returns the argument emphasised, if that is non-empty, otherwise it returns the null string. Italic corrections aren’t used, so this function should be used when punctuation will follow the result.

```
emphasize(s) ==
BEGIN
    if empty$(s) then return ""
    else return "{\em " * s * "}"
```

The ‘pop\$’ in this function gets rid of the duplicate ‘empty’ value and the ‘skip\$’ returns the duplicate field value

```
862 FUNCTION {field.or.null}
863 { duplicate$ empty$
864   { pop$ "" }
865   'skip$
866   if$
867 }
868
869 FUNCTION {emphasize}
870 { duplicate$ empty$
871   { pop$ "" }
872   { "\em{" swap$ * "}" * }
873   if$
874 }
875
876 FUNCTION {format.btitle}
877 { italic.book.title
878   entry.lang lang.en = and
879   'emphasize
880   'skip$
881   if$
882 }
883
```

#### B.4.1 Detect Language

```
884 INTEGERS { byte second.byte }
```

```

885
886 INTEGERS { char.lang tmp.lang }
887
888 STRINGS { tmp.str }
889
890 FUNCTION {get.str.lang}
891 { 'tmp.str :=
892   lang.other 'tmp.lang :=
893   #1 'charptr :=
894   tmp.str text.length$ #1 + 'len :=
895   { charptr len < }
896   { tmp.str charptr #1 substring$ chr.to.int$ 'byte :=
897     byte #128 <
898     { charptr #1 + 'charptr :=
899       byte #64 > byte #91 < and byte #96 > byte #123 < and or
900       { lang.en 'char.lang := }
901       { lang.other 'char.lang := }
902     if$
903   }
904   { tmp.str charptr #1 + #1 substring$ chr.to.int$ 'second.byte :=
905     byte #224 <

```

俄文西里尔字母: U+0400 到 U+052F, 对应 UTF-8 从 D0 80 到 D4 AF。

```

906     { charptr #2 + 'charptr :=
907       byte #207 > byte #212 < and
908       byte #212 = second.byte #176 < and or
909       { lang.ru 'char.lang := }
910       { lang.other 'char.lang := }
911     if$
912   }
913   { byte #240 <

```

CJK Unified Ideographs: U+4E00–U+9FFF; UTF-8: E4 B8 80–E9 BF BF.

```

914     { charptr #3 + 'charptr :=
915       byte #227 > byte #234 < and
916       { lang.zh 'char.lang := }

```

CJK Unified Ideographs Extension A: U+3400–U+4DBF; UTF-8: E3 90 80–E4 B6 BF.

```

917     { byte #227 =
918       { second.byte #143 >
919         { lang.zh 'char.lang := }

```

日语假名: U+3040–U+30FF, UTF-8: E3 81 80–E3 83 BF.

```

920     { second.byte #128 > second.byte #132 < and
921       { lang.ja 'char.lang := }
922       { lang.other 'char.lang := }
923     if$
924   }
925   if$
926 }

```

CJK Compatibility Ideographs: U+F900–U+FAFF, UTF-8: EF A4 80–EF AB BF.

```

927     { byte #239 =
928       second.byte #163 > second.byte #172 < and and

```



```

929             { lang.zh 'char.lang := }
930             { lang.other 'char.lang := }
931         if$
932     }
933     if$
934 }
935 if$
936 }

```

CJK Unified Ideographs Extension B–F: U+20000–U+2EBEF, UTF-8: F0 A0 80 80–F0 AE AF AF. CJK Compatibility Ideographs Supplement: U+2F800–U+2FA1F, UTF-8: F0 AF A0 80–F0 AF A8 9F.

```

937     { charptr #4 + 'charptr :=
938     byte #240 = second.byte #159 > and
939     { lang.zh 'char.lang := }
940     { lang.other 'char.lang := }
941     if$
942     }
943     if$
944     }
945     if$
946     }
947     if$
948     char.lang tmp.lang >
949     { char.lang 'tmp.lang := }
950     'skip$
951     if$
952     }
953 while$
954 tmp.lang
955 }
956
957 FUNCTION {check.entry.lang}
958 { author field.or.null
959   title field.or.null *
960   get.str.lang
961 }
962
963 STRINGS { entry.langid }
964
965 FUNCTION {set.entry.lang}
966 { "" 'entry.langid :=
967   language empty$ not
968   { language 'entry.langid := }
969   'skip$
970   if$
971   langid empty$ not
972   { langid 'entry.langid := }
973   'skip$
974   if$
975   entry.langid empty$
976   { check.entry.lang }
977   { entry.langid "english" = entry.langid "american" = or entry.langid "british" = or
978     { lang.en }

```

```

979         { entry.langid "chinese" =
980             { lang.zh }
981             { entry.langid "japanese" =
982                 { lang.ja }
983                 { entry.langid "russian" =
984                     { lang.ru }
985                     { check.entry.lang }
986                 if$
987             }
988             if$
989         }
990         if$
991     }
992     if$
993 }
994 if$
995 'entry.lang :=
996 }
997
998 FUNCTION {set.entry.numbered}
999 { type$ "patent" =
1000   type$ "standard" = or
1001   type$ "techreport" = or
1002   { #1 'entry.numbered := }
1003   { #0 'entry.numbered := }
1004   if$
1005 }
1006

```

#### B.4.2 Format names

The `format.names` function formats the argument (which should be in BibTeX name format) into "First Von Last, Junior", separated by commas and with an "and" before the last (but ending with "et al." if the last of multiple authors is "others"). This function's argument should always contain at least one name.

```

VAR: nameptr, namesleft, numnames: INTEGER
pseudoVAR: namesresult: STRING      (it's what's accumulated on the stack)

format.names(s) ==
BEGIN
    nameptr := 1
    numnames := num.names$(s)
    namesleft := numnames
    while namesleft > 0
    do
        % for full names:
        t := format.name$(s, nameptr, "{ff~}{vv~}{ll}{, jj}")
        % for abbreviated first names:
        t := format.name$(s, nameptr, "{f.~}{vv~}{ll}{, jj}")
        if nameptr > 1 then
            if namesleft > 1 then namesresult := namesresult * ", " * t
            else if numnames > 2
                then namesresult := namesresult * ", "

```

```

        fi
        if t = "others"
        then namerresult := namerresult * " et~al."
        else namerresult := namerresult * " and " * t
        fi
    fi
    else namerresult := t
    fi
    nameptr := nameptr + 1
    namesleft := namesleft - 1
od
return namerresult
END

```

The `format.authors` function returns the result of `format.names(author)` if the author is present, or else it returns the null string

```

format.authors ==
BEGIN
    if empty$(author) then return ""
    else return format.names(author)
    fi
END

```

`Format.editors` is like `format.authors`, but it uses the editor field, and appends ", editor" or ", editors"

```

format.editors ==
BEGIN
    if empty$(editor) then return ""
    else
        if num.names$(editor) > 1 then
            return format.names(editor) * ", editors"
        else
            return format.names(editor) * ", editor"
        fi
    fi
END

```

Other formatting functions are similar, so no "comment version" will be given for them.

```

1007 INTEGERS { nameptr namesleft numnames name.lang }
1008
1009 FUNCTION {format.name}
1010 { "{vv~}{ll}{, jj}{, ff}" format.names$ 't :=
1011   t "others" =
1012   { bbl.et.al }
1013   { t get.str.lang 'name.lang :=
1014     name.lang lang.en =
1015     { t #1 "{vv~}{ll}{ f{~}}" format.names$
1016       uppercase.name
1017       { "u" change.case$ }
1018       'skip$

```

```

1019         if$
1020         t #1 "{, jj}" format.name$ *
1021     }
1022     { t #1 "{ll}{ff}" format.name$ }
1023     if$
1024 }
1025 if$
1026 }
1027
1028 FUNCTION {format.names}
1029 { 's :=
1030   #1 'nameptr :=
1031   s num.names$ 'numnames :=
1032   ""
1033   numnames 'namesleft :=
1034   { namesleft #0 > }
1035   { s nameptr format.name bbl.et.al =
1036     numnames bibliography.et.al.min #1 - > nameptr bibliography.et.al.use.first > and or
1037     { ", " *
1038       bbl.et.al *
1039       #1 'namesleft :=
1040     }
1041     { nameptr #1 >
1042       { namesleft #1 = bbl.and "" = not and
1043         { bbl.and * }
1044         { ", " * }
1045       if$
1046     }
1047     'skip$
1048     if$
1049     s nameptr format.name *
1050   }
1051   if$
1052   nameptr #1 + 'nameptr :=
1053   namesleft #1 - 'namesleft :=
1054 }
1055 while$
1056 }
1057
1058 FUNCTION {format.key}
1059 { empty$
1060   { key field.or.null }
1061   { "" }
1062   if$
1063 }
1064
1065 FUNCTION {format.authors}
1066 { author empty$ not
1067   { author format.names }
1068   { "empty author in " cite$ * warning$
1069     (*author-year)
1070     bbl.anonymous
1071     (/author-year)
1072     (*numerical)
1073     ""

```

```

1074 </numerical>
1075     }
1076     if$
1077 }
1078
1079 FUNCTION {format.editors}
1080 { editor empty$
1081     { "" }
1082     { editor format.names }
1083     if$
1084 }
1085
1086 FUNCTION {format.translators}
1087 { translator empty$
1088     { "" }
1089     { translator format.names
1090         entry.lang lang.zh =
1091         { translator num.names$ #3 >
1092             { " 译" * }
1093             { ", 译" * }
1094             if$
1095         }
1096         'skip$
1097         if$
1098     }
1099     if$
1100 }
1101
1102 FUNCTION {format.full.names}
1103 { 's :=
1104     #1 'nameptr :=
1105     s num.names$ 'numnames :=
1106     numnames 'namesleft :=
1107     { namesleft #0 > }
1108     { s nameptr "{vv~}{ll}{, jj}{, ff}" format.name$ 't :=
1109         t get.str.lang 'name.lang :=
1110         name.lang lang.en =
1111         { t #1 "{vv~}{ll}" format.name$ 't := }
1112         { t #1 "{ll}{ff}" format.name$ 't := }
1113         if$
1114         nameptr #1 >
1115         {
1116             namesleft #1 >
1117             { ", " * t * }
1118             {
1119                 numnames #2 >
1120                 { ", " * }
1121                 'skip$
1122                 if$
1123                 t "others" =
1124                 { " et~al." * }
1125                 { " and " * t * }
1126                 if$
1127             }
1128             if$

```

```

1129     }
1130     't
1131     if$
1132     nameptr #1 + 'nameptr :=
1133     namesleft #1 - 'namesleft :=
1134     }
1135 while$
1136 }
1137
1138 FUNCTION {author.editor.full}
1139 { author empty$
1140   { editor empty$
1141     { "" }
1142     { editor format.full.names }
1143   if$
1144   }
1145   { author format.full.names }
1146 if$
1147 }
1148
1149 FUNCTION {author.full}
1150 { author empty$
1151   { "" }
1152   { author format.full.names }
1153 if$
1154 }
1155
1156 FUNCTION {editor.full}
1157 { editor empty$
1158   { "" }
1159   { editor format.full.names }
1160 if$
1161 }
1162
1163 FUNCTION {make.full.names}
1164 { type$ "book" =
1165   type$ "inbook" =
1166   or
1167   'author.editor.full
1168   { type$ "collection" =
1169     type$ "proceedings" =
1170     or
1171     'editor.full
1172     'author.full
1173   if$
1174   }
1175 if$
1176 }
1177
1178 FUNCTION {output.bibitem}
1179 { newline$
1180   "\bibitem[" writes$
1181   label ")" *
1182   make.full.names duplicates$ short.list =
1183   { pop$ }

```

```

1184 { duplicate$ "]" contains
1185   { "{" swap$ * "}" * }
1186   'skip$
1187   if$
1188   *
1189 }
1190 if$
1191 "]" * write$
1192 cite$ write$
1193 "]" write$
1194 newline$
1195 ""
1196 before.all 'output.state :=
1197 }
1198

```

### B.4.3 Format title

The `format.title` function is used for non-book-like titles. For most styles we convert to lowercase (except for the very first letter, and except for the first one after a colon (followed by whitespace)), and hope the user has brace-surrounded words that need to stay capitalized; for some styles, however, we leave it as it is in the database.

```

1199 FUNCTION {change.sentence.case}
1200 { entry.lang lang.en =
1201   { "t" change.case$ }
1202   'skip$
1203   if$
1204 }
1205
1206 FUNCTION {add.link}
1207 { url empty$ not
1208   { "\href{" url * "}" * swap$ * "}" * }
1209   { doi empty$ not
1210     { "\href{http://dx.doi.org/" doi * "}" * swap$ * "}" * }
1211     'skip$
1212     if$
1213   }
1214   if$
1215 }
1216
1217 FUNCTION {format.title}
1218 { title empty$
1219   { "" }
1220   { title
1221     sentence.case.title
1222     'change.sentence.case
1223     'skip$
1224     if$
1225     entry.numbered number empty$ not and
1226     { bbl.colon * number * }
1227     'skip$
1228     if$

```

```

1229      link.title
1230      'add.link
1231      'skip$
1232      if$
1233    }
1234  if$
1235 }
1236

```

For several functions we'll need to connect two strings with a tie (~) if the second one isn't very long (fewer than 3 characters). The `tie.or.space.connect` function does that. It concatenates the two strings on top of the stack, along with either a tie or space between them, and puts this concatenation back onto the stack:

```

tie.or.space.connect(str1,str2) ==
  BEGIN
    if text.length$(str2) < 3
      then return the concatenation of str1, "~", and str2
      else return the concatenation of str1, " ", and str2
    END

```

```

1237 FUNCTION {tie.or.space.connect}
1238 { duplicate$ text.length$ #3 <
1239   { "~" }
1240   { " " }
1241   if$
1242   swap$ * *
1243 }
1244

```

The `either.or.check` function complains if both fields or an either-or pair are nonempty.

```

either.or.check(t,s) ==
  BEGIN
    if empty$(s) then
      warning$(can't use both " * t * " fields in " * cite$)
    fi
  END

```

```

1245 FUNCTION {either.or.check}
1246 { empty$
1247   'pop$
1248   { "can't use both " swap$ * " fields in " * cite$ * warning$ }
1249   if$
1250 }
1251

```

The `format.bvolume` function is for formatting the volume and perhaps series name of a multivolume work. If both a volume and a series field are there, we assume the series field is the title of the whole multivolume work (the title field should be the title of the thing being referred to), and we add an "of <series>". This function is called in mid-sentence.



The `format.number.series` function is for formatting the series name and perhaps number of a work in a series. This function is similar to `format.bvolume`, although for this one the series must exist (and the volume must not exist). If the number field is empty we output either the series field unchanged if it exists or else the null string. If both the number and series fields are there we assume the series field gives the name of the whole series (the title field should be the title of the work being one referred to), and we add an "in <series>". We capitalize Number when this function is used at the beginning of a block.

```

1252 FUNCTION {is.digit}
1253 { duplicate$ empty$
1254   { pop$ #0 }
1255   { chr.to.int$
1256     duplicate$ "0" chr.to.int$ <
1257     { pop$ #0 }
1258     { "9" chr.to.int$ >
1259       { #0 }
1260       { #1 }
1261     if$
1262   }
1263 if$
1264 }
1265 if$
1266 }
1267
1268 FUNCTION {is.number}
1269 { 's :=
1270   s empty$
1271   { #0 }
1272   { s text.length$ 'charptr :=
1273     { charptr #0 >
1274       s charptr #1 substring$ is.digit
1275       and
1276     }
1277     { charptr #1 - 'charptr := }
1278     while$
1279     charptr not
1280   }
1281 if$
1282 }
1283
1284 FUNCTION {format.volume}
1285 { volume empty$ not
1286   { volume is.number
1287     { entry.lang lang.zh =
1288       { " 第 " volume * " 卷" * }
1289       { "volume" volume tie.or.space.connect }
1290     if$
1291   }
1292   { volume }
1293 if$
1294 }
```

```

1295     { "" }
1296   if$
1297 }
1298
1299 FUNCTION {format.number}
1300 { number empty$ not
1301   { number is.number
1302     { entry.lang lang.zh =
1303       { " 第 " number * " 册" * }
1304       { "number" number tie.or.space.connect }
1305       if$
1306     }
1307     { number }
1308   } if$
1309 }
1310 { "" }
1311 if$
1312 }
1313
1314 FUNCTION {format.volume.number}
1315 { volume empty$ not
1316   { format.volume }
1317   { format.number }
1318   if$
1319 }
1320
1321 FUNCTION {format.title.vol.num}
1322 { title
1323   sentence.case.title
1324   'change.sentence.case
1325   'skip$
1326   if$
1327   entry.numbered
1328   { number empty$ not
1329     { bbl.colon * number * }
1330     'skip$
1331     if$
1332   }
1333   { format.volume.number 's :=
1334     s empty$ not
1335     { bbl.colon * s * }
1336     'skip$
1337     if$
1338   }
1339   if$
1340 }
1341
1342 FUNCTION {format.series.vol.num.title}
1343 { format.volume.number 's :=
1344   series empty$ not
1345   { series
1346     sentence.case.title
1347     'change.sentence.case
1348     'skip$
1349     if$

```

```

1350     entry.numbered
1351         { bbl.wide.space * }
1352         { bbl.colon *
1353           s empty$ not
1354             { s * bbl.wide.space * }
1355           'skip$
1356         if$
1357       }
1358     if$
1359     title *
1360     sentence.case.title
1361       'change.sentence.case
1362       'skip$
1363     if$
1364     entry.numbered number empty$ not and
1365       { bbl.colon * number * }
1366       'skip$
1367     if$
1368   }
1369   { format.title.vol.num }
1370   if$
1371   format.btitle
1372   link.title
1373     'add.link
1374     'skip$
1375   if$
1376 }
1377
1378 FUNCTION {format.booktitle.vol.num}
1379 { booktitle
1380   entry.numbered
1381     'skip$
1382     { format.volume.number 's :=
1383       s empty$ not
1384         { bbl.colon * s * }
1385       'skip$
1386     if$
1387   }
1388   if$
1389 }
1390
1391 FUNCTION {format.series.vol.num.booktitle}
1392 { format.volume.number 's :=
1393   series empty$ not
1394     { series bbl.colon *
1395       entry.numbered not s empty$ not and
1396         { s * bbl.wide.space * }
1397       'skip$
1398     if$
1399     booktitle *
1400   }
1401   { format.booktitle.vol.num }
1402   if$
1403   format.btitle
1404   in.booktitle

```

```

1405     { duplicate$ empty$ not entry.lang lang.en = and
1406       { "In: " swap$ * }
1407       'skip$
1408     if$
1409   }
1410   'skip$
1411 if$
1412 }
1413
1414 FUNCTION {remove.period}
1415 { 't :=
1416   "" 's :=
1417   { t empty$ not }
1418   { t #1 #1 substring$ 'tmp.str :=
1419     tmp.str "." = not
1420     { s tmp.str * 's := }
1421     'skip$
1422     if$
1423     t #2 global.max$ substring$ 't :=
1424   }
1425   while$
1426   s
1427 }
1428
1429 FUNCTION {abbreviate}
1430 { remove.period
1431   't :=
1432   t "l" change.case$ 's :=
1433   ""
1434   s "physical review letters" =
1435   { "Phys Rev Lett" }
1436   'skip$
1437   if$
1438   's :=
1439   s empty$
1440   { t }
1441   { pop$ s }
1442   if$
1443 }
1444
1445 FUNCTION {format.journal}
1446 { ""
1447   short.journal
1448   { shortjournal empty$ not
1449     { shortjournal * }
1450     { journal empty$ not
1451       { journal * abbreviate }
1452       { journaltitle empty$ not
1453         { journaltitle * abbreviate }
1454         'skip$
1455         if$
1456       }
1457     }
1458     if$
1459   }

```

```

1460     }
1461     { journal empty$ not
1462       { journal * }
1463       { journaltitle empty$ not
1464         { journaltitle * }
1465         'skip$
1466         if$
1467       }
1468     if$
1469   }
1470   if$
1471   duplicate$ empty$ not
1472   { italic.journal entry.lang lang.en = and
1473     'emphasize
1474     'skip$
1475     if$
1476   }
1477   'skip$
1478   if$
1479 }
1480

```

#### B.4.4 Format entry type mark

```

1481 FUNCTION {set.entry.mark}
1482 { entry.mark empty$ not
1483   'pop$
1484   { mark empty$ not
1485     { pop$ mark 'entry.mark := }
1486     { 'entry.mark := }
1487     if$
1488   }
1489   if$
1490 }
1491
1492 FUNCTION {format.mark}
1493 { show.mark
1494   { entry.mark
1495     show.medium.type
1496     { medium empty$ not
1497       { "/" * medium * }
1498       { entry.is.electronic
1499         { "OL" * }
1500         'skip$
1501         if$
1502       }
1503     if$
1504   }
1505   'skip$
1506   if$
1507   'entry.mark :=
1508   space.before.mark
1509   { " " }
1510   { "\allowbreak" }
1511   if$

```

```

1512     "[" * entry.mark * "]" *
1513   }
1514   { "" }
1515   if$
1516 }
1517

```

#### B.4.5 Format edition

The `format.edition` function appends " edition" to the edition, if present. We lowercase the edition (it should be something like "Third"), because this doesn't start a sentence.

```

1518 FUNCTION {num.to.ordinal}
1519 { duplicate$ text.length$ 'charptr :=
1520   duplicate$ charptr #1 substring$ 's :=
1521   s "1" =
1522     { "st" * }
1523     { s "2" =
1524       { "nd" * }
1525       { s "3" =
1526         { "rd" * }
1527         { "th" * }
1528         if$
1529       }
1530     } if$
1531   }
1532   if$
1533 }
1534
1535 FUNCTION {format.edition}
1536 { edition empty$
1537   { "" }
1538   { edition is.number
1539     { entry.lang lang.zh =
1540       { edition " 版" * }
1541       { edition num.to.ordinal " ed." * }
1542     } if$
1543   }
1544   { entry.lang lang.en =
1545     { edition change.sentence.case 's :=
1546       s "Revised" = s "Revised edition" = or
1547       { "Rev. ed." }
1548       { s " ed." *}
1549     } if$
1550   }
1551   { edition }
1552   if$
1553 }
1554   if$
1555 }
1556   if$
1557 }
1558

```

#### B.4.6 Format publishing items

出版地址和出版社会有“[S.l.: s.n.]”的情况，所以必须一起处理。

```
1559 FUNCTION {format.publisher}
1560 { publisher empty$ not
1561   { publisher }
1562   { school empty$ not
1563     { school }
1564     { organization empty$ not
1565       { organization }
1566       { institution empty$ not
1567         { institution }
1568         { "" }
1569       if$
1570     }
1571   if$
1572 }
1573 if$
1574 }
1575 if$
1576 }
1577
1578 FUNCTION {format.address.publisher}
1579 { address empty$ not
1580   { address }
1581   { location empty$ not
1582     { location }
1583     { "" }
1584   if$
1585 }
1586 if$
1587 duplicates$ empty$ not
1588 { format.publisher empty$ not
1589   { bbl.colon * format.publisher * }
1590   { entry.is.electronic not show.missing.address.publisher and
1591     { bbl.colon * bbl.sine.nomine * }
1592     'skip$
1593   if$
1594 }
1595 if$
1596 }
1597 { pop$
1598   entry.is.electronic not show.missing.address.publisher and
1599   { format.publisher empty$ not
1600     { bbl.sine.loco bbl.colon * format.publisher * }
1601     { bbl.sine.loco.sine.nomine }
1602   if$
1603 }
1604 { format.publisher empty$ not
1605   { format.publisher }
1606   { "" }
1607 if$
1608 }
1609 if$
```

```

1610     }
1611     if$
1612 }
1613

```

#### B.4.7 Format date

The `format.date` function is for the month and year, but we give a warning if there's an empty year but the month is there, and we return the empty string if they're both empty.

期刊需要著录起止范围，其中年份使用“/”分隔，卷和期使用“-”分隔。版本 v2.0.2 前的年份也使用“-”分隔，仅提供兼容性，不再推荐。

```

1614 FUNCTION {extract.before.dash}
1615 { duplicate$ empty$
1616   { pop$ "" }
1617   { 's :=
1618     #1 'charptr :=
1619     s text.length$ #1 + 'len :=
1620     { charptr len <
1621       s charptr #1 substring$ "-" = not
1622       and
1623     }
1624     { charptr #1 + 'charptr := }
1625     while$
1626     s #1 charptr #1 - substring$
1627   }
1628   if$
1629 }
1630
1631 FUNCTION {extract.after.dash}
1632 { duplicate$ empty$
1633   { pop$ "" }
1634   { 's :=
1635     #1 'charptr :=
1636     s text.length$ #1 + 'len :=
1637     { charptr len <
1638       s charptr #1 substring$ "-" = not
1639       and
1640     }
1641     { charptr #1 + 'charptr := }
1642     while$
1643     { charptr len <
1644       s charptr #1 substring$ "-" =
1645       and
1646     }
1647     { charptr #1 + 'charptr := }
1648     while$
1649     s charptr global.max$ substring$
1650   }
1651   if$
1652 }
1653

```



```

1654 FUNCTION {extract.before.slash}
1655 { duplicate$ empty$
1656   { pop$ "" }
1657   { 's :=
1658     #1 'charptr :=
1659     s text.length$ #1 + 'len :=
1660     { charptr len <
1661       s charptr #1 substring$ "/" = not
1662       and
1663     }
1664     { charptr #1 + 'charptr := }
1665     while$
1666     s #1 charptr #1 - substring$
1667   }
1668   if$
1669 }
1670
1671 FUNCTION {extract.after.slash}
1672 { duplicate$ empty$
1673   { pop$ "" }
1674   { 's :=
1675     #1 'charptr :=
1676     s text.length$ #1 + 'len :=
1677     { charptr len <
1678       s charptr #1 substring$ "-" = not
1679       and
1680       s charptr #1 substring$ "/" = not
1681       and
1682     }
1683     { charptr #1 + 'charptr := }
1684     while$
1685     { charptr len <
1686       s charptr #1 substring$ "-" =
1687       s charptr #1 substring$ "/" =
1688       or
1689       and
1690     }
1691     { charptr #1 + 'charptr := }
1692     while$
1693     s charptr global.max$ substring$
1694   }
1695   if$
1696 }
1697

```

著者-出版年制必须提取出年份

```

1698 FUNCTION {format.year}
1699 { year empty$ not
1700   { year extract.before.slash extra.label * }
1701   { date empty$ not
1702     { date extract.before.dash extra.label * }
1703     { "empty year in " cite$ * warning$
1704       urldate empty$ not
1705       { "[" urldate extract.before.dash * extra.label * "]" * }
1706       { "" }

```

```

1707         if$
1708     }
1709     if$
1710 }
1711 if$
1712 }
1713
1714 FUNCTION {format.periodical.year}
1715 { year empty$ not
1716   { year extract.before.slash
1717     "--" *
1718     year extract.after.slash
1719     duplicate$ empty$
1720     'pop$
1721     { * }
1722     if$
1723   }
1724   { date empty$ not
1725     { date extract.before.dash }
1726     { "empty year in " cite$ * warning$
1727       urldate empty$ not
1728       { "[" urldate extract.before.dash * "]" * }
1729       { "" }
1730       if$
1731     }
1732     if$
1733   }
1734   if$
1735 }
1736

```

专利和报纸都是使用日期而不是年

```

1737 FUNCTION {format.date}
1738 { date empty$ not
1739   { type$ "patent" = type$ "newspaper" = or
1740     { date }
1741     { format.year }
1742     if$
1743   }
1744   { year empty$ not
1745     { format.year }
1746     { "" }
1747     if$
1748   }
1749   if$
1750 }
1751

```

更新、修改日期只用于电子资源 electronic

```

1752 FUNCTION {format.editdate}
1753 { date empty$ not
1754   { "\allowbreak(" date * ")" * }
1755   { "" }
1756   if$
1757 }

```

1758

国标中的“引用日期”都是与 URL 同时出现的，所以其实为 `urldate`，这个虽然不是 BibTeX 标准的域，但是实际中很常见。

```
1759 FUNCTION {format.urldate}
1760 { urldate empty$ not
1761   show.urldate show.url and is.pure.electronic or and
1762   url empty$ not and
1763   { "\allowbreak[" urldate * "]" * }
1764   { "" }
1765   if$
1766 }
1767
```

#### B.4.8 Format pages

By default, BibTeX sets the global integer variable `global.max$` to the BibTeX constant `glob_str_size`, the maximum length of a global string variable. Analogously, BibTeX sets the global integer variable `entry.max$` to `ent_str_size`, the maximum length of an entry string variable. The style designer may change these if necessary (but this is unlikely)

The `n.dashify` function makes each single ``-'` in a string a double ``--'` if it's not already

```
pseudoVAR: pageresult: STRING      (it's what's accumulated on the stack)

n.dashify(s) ==
BEGIN
  t := s
  pageresult := ""
  while (not empty$(t))
  do
    if (first character of t = "-")
    then
      if (next character isn't)
      then
        pageresult := pageresult * "--"
        t := t with the "-" removed
      else
        while (first character of t = "-")
        do
          pageresult := pageresult * "--"
          t := t with the "-" removed
        od
      fi
    else
      pageresult := pageresult * the first character
      t := t with the first character removed
    fi
  od
  return pageresult
```

END

国标里页码范围的连接号使用 hyphen，需要将 dash 转为 hyphen。

```
1768 FUNCTION {hyphenate}
1769 { 't :=
1770   ""
1771   { t empty$ not }
1772   { t #1 #1 substring$ "-" =
1773     { wave.dash.in.pages
1774       { "~" * }
1775       { "-" * }
1776       if$
1777       { t #1 #1 substring$ "-" = }
1778       { t #2 global.max$ substring$ 't := }
1779       while$
1780     }
1781     { t #1 #1 substring$ *
1782       t #2 global.max$ substring$ 't :=
1783     }
1784   if$
1785 }
1786 while$
1787 }
1788
```

This function doesn't begin a sentence so "pages" isn't capitalized. Other functions that use this should keep that in mind.

```
1789 FUNCTION {format.pages}
1790 { pages empty$
1791   { "" }
1792   { pages hyphenate }
1793   if$
1794 }
1795
1796 FUNCTION {format.extracted.pages}
1797 { pages empty$
1798   { "" }
1799   { pages
1800     only.start.page
1801     'extract.before.dash
1802     'hyphenate
1803     if$
1804   }
1805   if$
1806 }
1807
```

The format.vol.num.pages function is for the volume, number, and page range of a journal article. We use the format: vol(number):pages, with some variations for empty fields. This doesn't begin a sentence.

报纸在卷号缺失时，期号与前面的日期直接相连，所以必须拆开输出。

```
1808 FUNCTION {format.journal.volume}
```

```

1809 { volume empty$ not
1810   { bold.journal.volume
1811     { "\textbf{" volume * "}" * }
1812     { volume }
1813     if$
1814   }
1815   { "" }
1816   if$
1817 }
1818
1819 FUNCTION {format.journal.number}
1820 { number empty$ not
1821   { "\allowbreak (" number * ")" * }
1822   { "" }
1823   if$
1824 }
1825
1826 FUNCTION {format.journal.pages}
1827 { pages empty$
1828   { "" }
1829   { format.extracted.pages }
1830   if$
1831 }
1832

```

连续出版物的年卷期有起止范围，需要特殊处理

```

1833 FUNCTION {format.periodical.year.volume.number}
1834 { year empty$ not
1835   { year extract.before.slash }
1836   { "empty year in periodical " cite$ * warning$ }
1837   if$
1838   volume empty$ not
1839   { ", " * volume extract.before.dash * }
1840   'skip$
1841   if$
1842   number empty$ not
1843   { "\allowbreak (" * number extract.before.dash * ")" * }
1844   'skip$
1845   if$
1846   "--" *
1847   year extract.after.slash empty$
1848   volume extract.after.dash empty$ and
1849   number extract.after.dash empty$ and not
1850   { year extract.after.slash empty$ not
1851     { year extract.after.slash * }
1852     { year extract.before.slash * }
1853     if$
1854     volume empty$ not
1855     { ", " * volume extract.after.dash * }
1856     'skip$
1857     if$
1858     number empty$ not
1859     { "\allowbreak (" * number extract.after.dash * ")" * }
1860     'skip$
1861     if$

```

```

1862     }
1863     'skip$
1864   if$
1865 }
1866

```

#### B.4.9 Format url and doi

传统的 Bib<sub>T</sub><sub>E</sub><sub>X</sub> 习惯使用 howpublished 著录 url，这里提供支持。

```

1867 FUNCTION {check.url}
1868 { url empty$ not
1869   { "\url{" url * "}" * 'entry.url :=
1870     #1 'entry.is.electronic :=
1871   }
1872   { howpublished empty$ not
1873     { howpublished #1 #5 substring$ "\url{" =
1874       { howpublished 'entry.url :=
1875         #1 'entry.is.electronic :=
1876       }
1877       'skip$
1878     if$
1879   }
1880   { note empty$ not
1881     { note #1 #5 substring$ "\url{" =
1882       { note 'entry.url :=
1883         #1 'entry.is.electronic :=
1884       }
1885       'skip$
1886     if$
1887   }
1888   'skip$
1889   if$
1890 }
1891 if$
1892 }
1893 if$
1894 }
1895
1896 FUNCTION {format.url}
1897 { entry.url
1898 }
1899
1900 FUNCTION {output.url}
1901 { entry.url empty$ not
1902   { new.block
1903     entry.url output
1904   }
1905   'skip$
1906 if$
1907 }
1908

```

需要检测 DOI 是否已经包含在 URL 中。

```

1909 FUNCTION {check.doi}

```

```

1910 { doi empty$ not
1911   { #1 'entry.is.electronic := }
1912   'skip$
1913   if$
1914 }
1915
1916 FUNCTION {is.in.url}
1917 { 's :=
1918   s empty$
1919   { #1 }
1920   { entry.url empty$
1921     { #0 }
1922     { s text.length$ 'len :=
1923       entry.url text.length$ 'charptr :=
1924       { entry.url charptr len substring$ s = not
1925         charptr #0 >
1926         and
1927       }
1928       { charptr #1 - 'charptr := }
1929       while$
1930       charptr
1931     }
1932     if$
1933   }
1934   if$
1935 }
1936
1937 FUNCTION {format.doi}
1938 { ""
1939   doi empty$ not
1940   { "" 's :=
1941     doi 't :=
1942     #0 'numnames :=
1943     { t empty$ not}
1944     { t #1 #1 substring$ 'tmp.str :=
1945       tmp.str "," = tmp.str " " = or t #2 #1 substring$ empty$ or
1946       { t #2 #1 substring$ empty$
1947         { s tmp.str * 's := }
1948         'skip$
1949       }
1950       if$
1951       s empty$ s is.in.url or
1952       'skip$
1953       { numnames #1 + 'numnames :=
1954         numnames #1 >
1955         { ", " * }
1956         { "DOI: " * }
1957         if$
1958         "\doi{" s * "}" * *
1959       }
1960       if$
1961       "" 's :=
1962     }
1963     { s tmp.str * 's := }
1964     if$
1965     t #2 global.max$ substring$ 't :=

```

```

1965     }
1966     while$
1967 }
1968 'skip$
1969 if$
1970 }
1971
1972 FUNCTION {output.doi}
1973 { doi empty$ not show.doi and
1974   show.english.translation entry.lang lang.zh = and not and
1975   { new.block
1976     format.doi output
1977   }
1978   'skip$
1979   if$
1980 }
1981
1982 FUNCTION {check.electronic}
1983 { "" 'entry.url :=
1984   #0 'entry.is.electronic :=
1985   'check.doi
1986   'skip$
1987   if$
1988   'check.url
1989   'skip$
1990   if$
1991   medium empty$ not
1992   { medium "MT" = medium "DK" = or medium "CD" = or medium "OL" = or
1993     { #1 'entry.is.electronic := }
1994     'skip$
1995     if$
1996   }
1997   'skip$
1998   if$
1999 }
2000
2001 FUNCTION {format.eprint}
2002 { eprinttype empty$ not
2003   { eprinttype }
2004   { archivePrefix empty$ not
2005     { archivePrefix }
2006     { "" }
2007     if$
2008   }
2009   if$
2010   's :=
2011   s empty$ not
2012   { s ": \eprint{" *
2013     url empty$ not
2014     { url }
2015     { "https://" s "l" change.case$ * ".org/abs/" * eprint * }
2016     if$
2017     * "}" *
2018     eprint * "}" *
2019   }

```



```

2020     { eprint }
2021   if$
2022 }
2023
2024 FUNCTION {output.eprint}
2025 { show.preprint eprint empty$ not and
2026   { new.block
2027     format.eprint output
2028   }
2029   'skip$
2030   if$
2031 }
2032
2033 FUNCTION {format.note}
2034 { note empty$ not show.note and
2035   { note }
2036   { "" }
2037   if$
2038 }
2039
2040 FUNCTION {output.translation}
2041 { show.english.translation entry.lang lang.zh = and
2042   { translation empty$ not
2043     { translation }
2044     { "[English translation missing!]" }
2045     if$
2046     " (in Chinese)" * output
2047     write$
2048     format.doi duplicate$ empty$ not
2049     { newline$
2050       write$
2051     }
2052     'pop$
2053     if$
2054     " \\" write$
2055     newline$
2056     "(" write$
2057     ""
2058     before.all 'output.state :=
2059   }
2060   'skip$
2061   if$
2062 }
2063

```

The function `empty.misc.check` complains if all six fields are empty, and if there's been no sorting or alphabetic-label complaint.

```

2064 FUNCTION {empty.misc.check}
2065 { author empty$ title empty$
2066   year empty$
2067   and and
2068   key empty$ not and
2069   { "all relevant fields are empty in " cite$ * warning$ }
2070   'skip$

```

```

2071 if$
2072 }
2073

```

## B.5 Functions for all entry types

Now we define the type functions for all entry types that may appear in the .BIB file—e.g., functions like ‘article’ and ‘book’. These are the routines that actually generate the .BBL-file output for the entry. These must all precede the READ command. In addition, the style designer should have a function ‘default.type’ for unknown types. Note: The fields (within each list) are listed in order of appearance, except as described for an ‘inbook’ or a ‘proceedings’.

### B.5.1 专著

```

2074 FUNCTION {monograph}
2075 { output.bibitem
2076   output.translation
2077   author empty$ not
2078     { format.authors }
2079     { editor empty$ not
2080       { format.editors }
2081       { "empty author and editor in " cite$ * warning$
2082 (*author-year)
2083   bbl.anonymous
2084 (/author-year)
2085 (*numerical)
2086   ""
2087 (/numerical)
2088   }
2089   if$
2090   }
2091   if$
2092   output
2093   year.after.author
2094     { period.after.author
2095       'new.sentence
2096       'skip$
2097       if$
2098       format.year "year" output.check
2099     }
2100     'skip$
2101   if$
2102   new.block
2103   format.series.vol.num.title "title" output.check
2104   "M" set.entry.mark
2105   format.mark "" output.after
2106   new.block
2107   format.translators output
2108   new.sentence
2109   format.edition output

```

```

2110 new.block
2111 format.address.publisher output
2112 year.after.author not
2113   { format.year "year" output.check }
2114   'skip$
2115   if$
2116   format.pages bbl.pages.colon output.after
2117   format.urldate "" output.after
2118   output.url
2119   output.doi
2120 new.block
2121 format.note output
2122 fin.entry
2123 }
2124

```

## B.5.2 专著中的析出文献

An incollection is like inbook, but where there is a separate title for the referenced thing (and perhaps an editor for the whole). An incollection may CROSSREF a book.

Required: author, title, booktitle, publisher, year

Optional: editor, volume or number, series, type, chapter, pages, address, edition, month, note

```

2125 FUNCTION {incollection}
2126 { output.bibitem
2127   output.translation
2128   format.authors output
2129   author format.key output
2130   year.after.author
2131     { period.after.author
2132       'new.sentence
2133       'skip$
2134       if$
2135       format.year "year" output.check
2136     }
2137     'skip$
2138   if$
2139   new.block
2140   format.title "title" output.check
2141   "M" set.entry.mark
2142   format.mark "" output.after
2143   new.block
2144   format.translators output
2145   new.slash
2146   format.editors output
2147   new.block
2148   format.series.vol.num.booktitle "booktitle" output.check
2149   new.block
2150   format.edition output
2151   new.block
2152   format.address.publisher output

```

```

2153 year.after.author not
2154   { format.year "year" output.check }
2155   'skip$
2156 if$
2157 format.extracted.pages bbl.pages.colon output.after
2158 format.urldate "" output.after
2159 output.url
2160 output.doi
2161 new.block
2162 format.note output
2163 fin.entry
2164 }
2165

```

### B.5.3 连续出版物

```

2166 FUNCTION {periodical}
2167 { output.bibitem
2168   output.translation
2169   format.authors output
2170   author format.key output
2171   year.after.author
2172     { period.after.author
2173       'new.sentence
2174       'skip$
2175       if$
2176       format.year "year" output.check
2177     }
2178     'skip$
2179   if$
2180   new.block
2181   format.title "title" output.check
2182   "J" set.entry.mark
2183   format.mark "" output.after
2184   new.block
2185   format.periodical.year.volume.number output
2186   new.block
2187   format.address.publisher output
2188   year.after.author not
2189     { format.periodical.year "year" output.check }
2190     'skip$
2191   if$
2192   format.urldate "" output.after
2193   output.url
2194   output.doi
2195   new.block
2196   format.note output
2197   fin.entry
2198 }
2199

```

### B.5.4 连续出版物中的析出文献

The article function is for an article in a journal. An article may CROSSREF another article.

Required fields: author, title, journal, year

Optional fields: volume, number, pages, month, note

The other entry functions are all quite similar, so no "comment version" will be given for them.

```
2200 FUNCTION {article}
2201 { output.bibitem
2202   output.translation
2203   format.authors output
2204   author format.key output
2205   year.after.author
2206     { period.after.author
2207       'new.sentence
2208       'skip$
2209       if$
2210       format.year "year" output.check
2211     }
2212     'skip$
2213   if$
2214   new.block
2215   title.in.journal
2216     { format.title "title" output.check
2217       "J" set.entry.mark
2218       format.mark "" output.after
2219       new.block
2220     }
2221     'skip$
2222   if$
2223   format.journal "journal" output.check
2224   year.after.author not
2225     { format.date "year" output.check }
2226     'skip$
2227   if$
2228   format.journal.volume output
2229   format.journal.number "" output.after
2230   format.journal.pages bbl.pages.colon output.after
2231   format.urldate "" output.after
2232   output.url
2233   output.doi
2234   new.block
2235   format.note output
2236   fin.entry
2237 }
2238
```

### B.5.5 专利文献

number 域也可以用来表示专利号。

```
2239 FUNCTION {patent}
2240 { output.bibitem
2241   output.translation
2242   format.authors output
2243   author format.key output
```

```

2244 year.after.author
2245     { period.after.author
2246         'new.sentence
2247         'skip$
2248         if$
2249         format.year "year" output.check
2250     }
2251     'skip$
2252 if$
2253 new.block
2254 format.title "title" output.check
2255 "P" set.entry.mark
2256 format.mark "" output.after
2257 new.block
2258 format.date "year" output.check
2259 format.urldate "" output.after
2260 output.url
2261 output.doi
2262 new.block
2263 format.note output
2264 fin.entry
2265 }
2266

```

## B.5.6 电子资源

```

2267 FUNCTION {electronic}
2268 { #1 #1 check.electronic
2269   #1 'entry.is.electronic :=
2270   #1 'is.pure.electronic :=
2271   output.bibitem
2272   output.translation
2273   format.authors output
2274   author format.key output
2275   year.after.author
2276     { period.after.author
2277         'new.sentence
2278         'skip$
2279         if$
2280         format.year "year" output.check
2281     }
2282     'skip$
2283 if$
2284 new.block
2285 format.series.vol.num.title "title" output.check
2286 "EB" set.entry.mark
2287 format.mark "" output.after
2288 new.block
2289 format.address.publisher output
2290 year.after.author not
2291   { date empty$
2292     { format.date output }
2293     'skip$
2294   if$
2295 }

```

```

2296     'skip$
2297   if$
2298     format.pages bbl.pages.colon output.after
2299     format.editdate "" output.after
2300     format.urldate "" output.after
2301     output.url
2302     output.doi
2303     new.block
2304     format.note output
2305     fin.entry
2306 }
2307

```

### B.5.7 预印本

```

2308 FUNCTION {preprint}
2309 { output.bibitem
2310   output.translation
2311   author empty$ not
2312     { format.authors }
2313     { editor empty$ not
2314       { format.editors }
2315       { "empty author and editor in " cite$ * warning$
2316 (*author-year)
2317       bbl.anonymous
2318 (/author-year)
2319 (*numerical)
2320 ""
2321 (/numerical)
2322 }
2323 if$
2324 }
2325 if$
2326 output
2327 year.after.author
2328   { period.after.author
2329     'new.sentence
2330     'skip$
2331     if$
2332     format.year "year" output.check
2333   }
2334   'skip$
2335   if$
2336   new.block
2337   title.in.journal
2338     { format.series.vol.num.title "title" output.check
2339 (*!2005)
2340     "Z" set.entry.mark
2341 (/!2005)
2342 (*2005)
2343     "M" set.entry.mark
2344 (/2005)
2345     format.mark "" output.after
2346     new.block
2347   }
2348   'skip$

```

```

2349 if$
2350 format.translators output
2351 new.sentence
2352 format.edition output
2353 new.block
2354 output.eprint
2355 year.after.author not
2356   { format.year "year" output.check }
2357   'skip$
2358 if$
2359 format.pages bbl.pages.colon output.after
2360 format.urldate "" output.after
2361 output.url
2362 new.block
2363 format.note output
2364 fin.entry
2365 }
2366

```

### B.5.8 其他文献类型

A misc is something that doesn't fit elsewhere.

Required: at least one of the 'optional' fields

Optional: author, title, howpublished, month, year, note

Misc 用来自动判断类型。

```

2367 FUNCTION {misc}
2368 { journal empty$ not
2369   'article
2370   { booktitle empty$ not
2371     'incollection
2372     { publisher empty$ not
2373       'monograph
2374       { eprint empty$ not show.preprint and
2375         'preprint
2376         { entry.is.electronic
2377           'electronic
2378           {
2379             (*!2005)
2380               "Z" set.entry.mark
2381             (/!2005)
2382             (*2005)
2383               "M" set.entry.mark
2384             (/2005)
2385               monograph
2386             }
2387           }
2388         }
2389       }
2390     }
2391   }
2392 }
2393 if$
2394 }

```



```

2395 if$
2396 empty.misc.check
2397 }
2398
2399 FUNCTION {archive}
2400 { "A" set.entry.mark
2401   misc
2402 }
2403

```

The book function is for a whole book. A book may CROSSREF another book.

Required fields: author or editor, title, publisher, year

Optional fields: volume or number, series, address, edition, month, note

```

2404 FUNCTION {book} { monograph }
2405

```

A booklet is a bound thing without a publisher or sponsoring institution.

Required: title

Optional: author, howpublished, address, month, year, note

```

2406 FUNCTION {booklet} { book }
2407
2408 FUNCTION {collection}
2409 { "G" set.entry.mark
2410   monograph
2411 }
2412
2413 FUNCTION {database}
2414 { "DB" set.entry.mark
2415   electronic
2416 }
2417
2418 FUNCTION {dataset}
2419 { "DS" set.entry.mark
2420   electronic
2421 }
2422

```

An inbook is a piece of a book: either a chapter and/or a page range. It may CROSSREF a book. If there's no volume field, the type field will come before number and series.

Required: author or editor, title, chapter and/or pages, publisher, year

Optional: volume or number, series, type, address, edition, month, note

inbook 类是不含 booktitle 域的，所以不应该适用于“专著中的析出文献”，而应该是专著，即 book 类。

```

2423 FUNCTION {inbook} { book }
2424

```

An inproceedings is an article in a conference proceedings, and it may CROSSREF a proceedings. If there's no address field, the month (& year) will appear just

before note.

Required: author, title, booktitle, year

Optional: editor, volume or number, series, pages, address, month, organization,

publisher, note

```
2425 FUNCTION {inproceedings}
2426 { "C" set.entry.mark
2427   incollection
2428 }
2429
```

The conference function is included for Scribe compatibility.

```
2430 FUNCTION {conference} { inproceedings }
2431
2432 FUNCTION {map}
2433 { "CM" set.entry.mark
2434   misc
2435 }
2436
```

A manual is technical documentation.

Required: title

Optional: author, organization, address, edition, month, year, note

```
2437 FUNCTION {manual} { monograph }
2438
```

A mastersthesis is a Master's thesis.

Required: author, title, school, year

Optional: type, address, month, note

```
2439 FUNCTION {mastersthesis}
2440 { "D" set.entry.mark
2441   monograph
2442 }
2443
2444 FUNCTION {newspaper}
2445 { "N" set.entry.mark
2446   article
2447 }
2448
2449 FUNCTION {online}
2450 { "EB" set.entry.mark
2451   electronic
2452 }
2453
```

A phdthesis is like a mastersthesis.

Required: author, title, school, year

Optional: type, address, month, note

```
2454 FUNCTION {phdthesis} { mastersthesis }
2455
```

A proceedings is a conference proceedings. If there is an organization but no editor field, the organization will appear as the first optional field (we try to make the first block nonempty); if there's no address field, the month (& year) will appear just before note.

Required: title, year

Optional: editor, volume or number, series, address, month, organization, publisher, note

```
2456 FUNCTION {proceedings}
2457 { "C" set.entry.mark
2458   monograph
2459 }
2460
2461 FUNCTION {software}
2462 { "CP" set.entry.mark
2463   electronic
2464 }
2465
2466 FUNCTION {standard}
2467 { "S" set.entry.mark
2468   misc
2469 }
2470
```

A techreport is a technical report.

Required: author, title, institution, year

Optional: type, number, address, month, note

```
2471 FUNCTION {techreport}
2472 { "R" set.entry.mark
2473   misc
2474 }
2475
```

An unpublished is something that hasn't been published.

Required: author, title, note

Optional: month, year

```
2476 FUNCTION {unpublished} { misc }
2477
```

We use entry type 'misc' for an unknown type; BibTeX gives a warning.

```
2478 FUNCTION {default.type} { misc }
2479
```

## B.6 Common macros

Here are macros for common things that may vary from style to style. Users are encouraged to use these macros.

Months are either written out in full or abbreviated

2480 MACRO {jan} {"January"}  
 2481  
 2482 MACRO {feb} {"February"}  
 2483  
 2484 MACRO {mar} {"March"}  
 2485  
 2486 MACRO {apr} {"April"}  
 2487  
 2488 MACRO {may} {"May"}  
 2489  
 2490 MACRO {jun} {"June"}  
 2491  
 2492 MACRO {jul} {"July"}  
 2493  
 2494 MACRO {aug} {"August"}  
 2495  
 2496 MACRO {sep} {"September"}  
 2497  
 2498 MACRO {oct} {"October"}  
 2499  
 2500 MACRO {nov} {"November"}  
 2501  
 2502 MACRO {dec} {"December"}  
 2503

Journals are either written out in full or abbreviated; the abbreviations are like those found in ACM publications.

To get a completely different set of abbreviations, it may be best to make a separate .bib file with nothing but those abbreviations; users could then include that file name as the first argument to the \bibliography command

2504 MACRO {acmcs} {"ACM Computing Surveys"}  
 2505  
 2506 MACRO {acta} {"Acta Informatica"}  
 2507  
 2508 MACRO {cacm} {"Communications of the ACM"}  
 2509  
 2510 MACRO {ibmjrd} {"IBM Journal of Research and Development"}  
 2511  
 2512 MACRO {ibmsj} {"IBM Systems Journal"}  
 2513  
 2514 MACRO {ieeese} {"IEEE Transactions on Software Engineering"}  
 2515  
 2516 MACRO {ieeetc} {"IEEE Transactions on Computers"}  
 2517  
 2518 MACRO {ieeetcad}  
 2519 {"IEEE Transactions on Computer-Aided Design of Integrated Circuits"}  
 2520  
 2521 MACRO {ipl} {"Information Processing Letters"}  
 2522  
 2523 MACRO {jacm} {"Journal of the ACM"}  
 2524  
 2525 MACRO {jcss} {"Journal of Computer and System Sciences"}  
 2526

```

2527 MACRO {scp} {"Science of Computer Programming"}
2528
2529 MACRO {sicomp} {"SIAM Journal on Computing"}
2530
2531 MACRO {tocs} {"ACM Transactions on Computer Systems"}
2532
2533 MACRO {tods} {"ACM Transactions on Database Systems"}
2534
2535 MACRO {tog} {"ACM Transactions on Graphics"}
2536
2537 MACRO {toms} {"ACM Transactions on Mathematical Software"}
2538
2539 MACRO {toois} {"ACM Transactions on Office Information Systems"}
2540
2541 MACRO {toplas} {"ACM Transactions on Programming Languages and Systems"}
2542
2543 MACRO {tcs} {"Theoretical Computer Science"}
2544

```

## B.7 Format labels

The `sortify` function converts to lower case after purifying; it's used in sorting and in computing alphabetic labels after sorting

The `chop.word(w,len,s)` function returns either `s` or, if the first `len` letters of `s` equals `w` (this comparison is done in the third line of the function's definition), it returns that part of `s` after `w`.

```

2545 FUNCTION {sortify}
2546 { purify$
2547   "l" change.case$
2548 }
2549

```

We need the `chop.word` stuff for the dubious unsorted-list-with-labels case.

```

2550 FUNCTION {chop.word}
2551 { 's :=
2552   'len :=
2553   s #1 len substring$ =
2554     { s len #1 + global.max$ substring$ }
2555     's
2556   if$
2557 }
2558

```

The `format.lab.names` function makes a short label by using the initials of the von and Last parts of the names (but if there are more than four names, (i.e., people) it truncates after three and adds a superscripted "+"; it also adds such a "+" if the last of multiple authors is "others"). If there is only one name, and its von and Last parts combined have just a single name-token ("Knuth" has a single token, "Brinch Hansen" has two), we take the first three letters of the last name. The boolean

et.al.char.used tells whether we've used a superscripted "+", so that we know whether to include a LaTeX macro for it.

```
format.lab.names(s) ==
BEGIN
  numnames := num.names$(s)
  if numnames > 1 then
    if numnames > 4 then
      namesleft := 3
    else
      namesleft := numnames
    nameptr := 1
    nameresult := ""
    while namesleft > 0
      do
        if (name_ptr = numnames) and
          format.name$(s, nameptr, "{ff }{vv }{ll}{ jj}") = "others"
        then nameresult := nameresult * "{\etalchar{+}}"
          et.al.char.used := true
        else nameresult := nameresult *
          format.name$(s, nameptr, "{v}{l}")
          nameptr := nameptr + 1
          namesleft := namesleft - 1
        od
      if numnames > 4 then
        nameresult := nameresult * "{\etalchar{+}}"
        et.al.char.used := true
      else
        t := format.name$(s, 1, "{v}{l}")
        if text.length$(t) < 2 then % there's just one name-token
          nameresult := text.prefix$(format.name$(s,1,"{ll}"),3)
        else
          nameresult := t
        fi
      fi
    return nameresult
  END
```

Exactly what fields we look at in constructing the primary part of the label depends on the entry type; this selectivity (as opposed to, say, always looking at author, then editor, then key) helps ensure that "ignored" fields, as described in the LaTeX book, really are ignored. Note that MISC is part of the deepest 'else' clause in the nested part of calc.label; thus, any unrecognized entry type in the database is handled correctly.

There is one auxiliary function for each of the four different sequences of fields we use. The first of these functions looks at the author field, and then, if necessary, the key field. The other three functions, which might look at two fields and the key field, are similar, except that the key field takes precedence over the organization field (for labels—not for sorting).

The `calc.label` function calculates the preliminary label of an entry, which is formed by taking three letters of information from the author or editor or key or organization field (depending on the entry type and on what's empty, but ignoring a leading "The " in the organization), and appending the last two characters (digits) of the year. It is an error if the appropriate fields among author, editor, organization, and key are missing, and we use the first three letters of the `cite$` in desperation when this happens. The resulting label has the year part, but not the name part, `purify$ed` (`purify$ing` the year allows some sorting shenanigans by the user).

This function also calculates the version of the label to be used in sorting.

The final label may need a trailing 'a', 'b', etc., to distinguish it from otherwise identical labels, but we can't calculate those "extra.label"s until after sorting.

```
calc.label ==
BEGIN
  if type$ = "book" or "inbook" then
    author.editor.key.label
  else if type$ = "proceedings" then
    editor.key.organization.label
  else if type$ = "manual" then
    author.key.organization.label
  else
    author.key.label
  fi fi fi
  label := label * substring$(purify$(field.or.null(year)), -1, 2)
    % assuming we will also sort, we calculate a sort.label
  sort.label := sortify(label), but use the last four, not two, digits
END
```

```
2559 FUNCTION {format.lab.name}
2560 { "{vv~}{ll}{, jj}{, ff}" format.name$ 't :=
2561   t "others" =
2562     { citation.et.al }
2563     { t get.str.lang 'name.lang :=
2564       name.lang lang.zh = name.lang lang.ja = or
2565         { t #1 "{ll}{ff}" format.name$ }
2566         { t #1 "{vv~}{ll}" format.name$ }
2567       if$
2568     }
2569   if$
2570 }
2571
```

第一作者姓名相同、年份相同但作者数量不同时，也需要年份标签区分。比如“王临惠 等,2010a”和“王临惠,2010b”，所以使用 `short.label` 存储不带“et al”的版本。

```
2572 FUNCTION {format.lab.names}
2573 { 's :=
2574   s #1 format.lab.name 'short.label :=
2575   #1 'nameptr :=
```

```

2576 s num.names$ 'numnames :=
2577 ""
2578 numnames 'namesleft :=
2579 { namesleft #0 > }
2580 { s nameptr format.lab.name citation.et.al =
2581   numnames citation.et.al.min #1 - > nameptr citation.et.al.use.first > and or
2582   { bbl.space *
2583     citation.et.al *
2584     #1 'namesleft :=
2585   }
2586   { nameptr #1 >
2587     { namesleft #1 = citation.and "" = not and
2588       { citation.and * }
2589       { ", " * }
2590       if$
2591     }
2592     'skip$
2593     if$
2594     s nameptr format.lab.name *
2595   }
2596   if$
2597   nameptr #1 + 'nameptr :=
2598   namesleft #1 - 'namesleft :=
2599 }
2600 while$
2601 }
2602
2603 FUNCTION {author.key.label}
2604 { author empty$
2605   { key empty$
2606     { cite$ #1 #3 substring$ }
2607     'key
2608     if$
2609   }
2610   { author format.lab.names }
2611   if$
2612 }
2613
2614 FUNCTION {author.editor.key.label}
2615 { author empty$
2616   { editor empty$
2617     { key empty$
2618       { cite$ #1 #3 substring$ }
2619       'key
2620       if$
2621     }
2622     { editor format.lab.names }
2623     if$
2624   }
2625   { author format.lab.names }
2626   if$
2627 }
2628
2629 FUNCTION {author.key.organization.label}
2630 { author empty$

```



```

2631 { key empty$
2632   { organization empty$
2633     { cite$ #1 #3 substring$ }
2634     { "The " #4 organization chop.word #3 text.prefix$ }
2635     if$
2636   }
2637   'key
2638   if$
2639 }
2640 { author format.lab.names }
2641 if$
2642 }
2643
2644 FUNCTION {editor.key.organization.label}
2645 { editor empty$
2646   { key empty$
2647     { organization empty$
2648       { cite$ #1 #3 substring$ }
2649       { "The " #4 organization chop.word #3 text.prefix$ }
2650       if$
2651     }
2652     'key
2653     if$
2654   }
2655   { editor format.lab.names }
2656   if$
2657 }
2658
2659 FUNCTION {calc.short.authors}
2660 { "" 'short.label :=
2661   type$ "book" =
2662   type$ "inbook" =
2663   or
2664   'author.editor.key.label
2665   { type$ "collection" =
2666     type$ "proceedings" =
2667     or
2668     { editor empty$ not
2669       'editor.key.organization.label
2670       'author.key.organization.label
2671       if$
2672     }
2673     'author.key.label
2674     if$
2675   }
2676   if$
2677   'short.list :=
2678   short.label empty$
2679   { short.list 'short.label := }
2680   'skip$
2681   if$
2682 }
2683

```

如果 label 中有中括号“[”，分别用大括号保护起来，防止 \bibitem 处理出

错。另外为了兼容 `bibunits`, “`name(year)fullname`”的每一项都要分别保护起来, 参考 [tuna/thuthesis/#630](https://tuna.thuthesis.org/#630)。

```
2684 FUNCTION {calc.label}
2685 { calc.short.authors
2686   short.list "]" contains
2687   { "{" short.list * "]" * }
2688   { short.list }
2689   if$
2690   "("
2691   *
2692   format.year duplicate$ empty$
2693   short.list key field.or.null = or
2694   { pop$ "" }
2695   'skip$
2696   if$
2697   duplicate$ "]" contains
2698   { "{" swap$ * "]" * }
2699   'skip$
2700   if$
2701   *
2702   'label :=
2703   short.label
2704   "("
2705   *
2706   format.year duplicate$ empty$
2707   short.list key field.or.null = or
2708   { pop$ "" }
2709   'skip$
2710   if$
2711   *
2712   'short.label :=
2713 }
2714
```

## B.8 Sorting

When sorting, we compute the sortkey by executing “presort” on each entry. The presort key contains a number of “sortify”ed strings, concatenated with multiple blanks between them. This makes things like “brinch per” come before “brinch hansen per”.

The fields used here are: the `sort.label` for alphabetic labels (as set by `calc.label`), followed by the author names (or editor names or organization (with a leading “The” removed) or key field, depending on entry type and on what’s empty), followed by year, followed by the first bit of the title (chopping off a leading “The”, “A”, or “An”). Names are formatted: Von Last First Junior. The names within a part will be separated by a single blank (such as “brinch hansen”), two will separate the name parts themselves (except the von and last), three will separate the names, four will separate the names from year (and from label, if alphabetic), and four will separate

year from title.

The `sort.format.names` function takes an argument that should be in BibTeX name format, and returns a string containing ” ”-separated names in the format described above. The function is almost the same as `format.names`.

```
2715 (*author-year)
2716 FUNCTION {sort.language.label}
2717 { entry.lang lang.zh =
2718   { lang.zh.order }
2719   { entry.lang lang.ja =
2720     { lang.ja.order }
2721     { entry.lang lang.en =
2722       { lang.en.order }
2723       { entry.lang lang.ru =
2724         { lang.ru.order }
2725         { lang.other.order }
2726         if$
2727       }
2728     } if$
2729   }
2730 } if$
2731 }
2732 if$
2733 #64 +
2734 int.to.chr$
2735 }
2736
2737 FUNCTION {sort.format.names}
2738 { 's :=
2739   #1 'nameptr :=
2740   ""
2741   s num.names$ 'numnames :=
2742   numnames 'namesleft :=
2743   { namesleft #0 > }
2744   {
2745     s nameptr "{vv{ } }{ll{ }}{ ff{ }}{ jj{ }}" format.name$ 't :=
2746     nameptr #1 >
2747     {
2748       " " *
2749       namesleft #1 = t "others" = and
2750       { "zzzzz" * }
2751       { numnames #2 > nameptr #2 = and
2752         { "zz" * year field.or.null * " " * }
2753         'skip$
2754       } if$
2755       t sortify *
2756     }
2757     if$
2758   }
2759   { t sortify * }
2760   if$
2761   nameptr #1 + 'nameptr :=
2762   namesleft #1 - 'namesleft :=
2763 }
```

```

2764 while$
2765 }
2766

```

The `sort.format.title` function returns the argument, but first any leading "A"s, "An"s, or "The"s are removed. The `chop.word` function uses `s`, so we need another string variable, `t`

```

2767 FUNCTION {sort.format.title}
2768 { 't :=
2769   "A " #2
2770   "An " #3
2771   "The " #4 t chop.word
2772   chop.word
2773   chop.word
2774   sortify
2775   #1 global.max$ substring$
2776 }
2777

```

The auxiliary functions here, for the `presort` function, are analogous to the ones for `calc.label`; the same comments apply, except that the organization field takes precedence here over the key field. For sorting purposes, we still remove a leading "The" from the organization field.

```

2778 FUNCTION {anonymous.sort}
2779 { entry.lang lang.zh =
2780   { "yi4 ming2" }
2781   { "anon" }
2782   if$
2783 }
2784
2785 FUNCTION {warn.empty.key}
2786 { entry.lang lang.zh =
2787   { "empty key in " cite$ * warning$ }
2788   'skip$
2789   if$
2790 }
2791
2792 FUNCTION {author.sort}
2793 { key empty$
2794   { warn.empty.key
2795     author empty$
2796     { anonymous.sort }
2797     { author sort.format.names }
2798     if$
2799   }
2800   { key }
2801   if$
2802 }
2803
2804 FUNCTION {author.editor.sort}
2805 { key empty$
2806   { warn.empty.key

```

```

2807     author empty$
2808     { editor empty$
2809         { anonymous.sort }
2810         { editor sort.format.names }
2811     if$
2812     }
2813     { author sort.format.names }
2814 if$
2815 }
2816 { key }
2817 if$
2818 }
2819
2820 FUNCTION {author.organization.sort}
2821 { key empty$
2822   { warn.empty.key
2823     author empty$
2824     { organization empty$
2825       { anonymous.sort }
2826       { "The " #4 organization chop.word sortify }
2827     if$
2828   }
2829   { author sort.format.names }
2830 if$
2831 }
2832 { key }
2833 if$
2834 }
2835
2836 FUNCTION {editor.organization.sort}
2837 { key empty$
2838   { warn.empty.key
2839     editor empty$
2840     { organization empty$
2841       { anonymous.sort }
2842       { "The " #4 organization chop.word sortify }
2843     if$
2844   }
2845   { editor sort.format.names }
2846 if$
2847 }
2848 { key }
2849 if$
2850 }
2851
2852 </author-year>

```

顺序编码制的排序要简单得多

```

2853 <*numerical>
2854 INTEGERS { seq.num }
2855
2856 FUNCTION {init.seq}
2857 { #0 'seq.num :=}
2858
2859 FUNCTION {int.to.fix}

```

```

2860 { "00000000" swap$ int.to.str$ *
2861 #-1 #10 substring$
2862 }
2863
2864 </numerical>

```

There is a limit, `entry.max$`, on the length of an entry string variable (which is what its `sort.key$` is), so we take at most that many characters of the constructed key, and hope there aren't many references that match to that many characters!

```

2865 FUNCTION {presort}
2866 { set.entry.lang
2867   set.entry.numbered
2868   show.url show.doi check.electronic
2869   #0 'is.pure.electronic :=
2870   calc.label
2871   label sortify
2872   " "
2873   *
2874   <*author-year>
2875   sort.language.label
2876   " "
2877   *
2878   type$ "book" =
2879   type$ "inbook" =
2880   or
2881     'author.editor.sort
2882     { type$ "collection" =
2883       type$ "proceedings" =
2884       or
2885         'editor.organization.sort
2886         'author.sort
2887       if$
2888     }
2889   if$
2890   *
2891   " "
2892   *
2893   year field.or.null sortify
2894   *
2895   " "
2896   *
2897   cite$
2898   *
2899   #1 entry.max$ substring$
2900   </author-year>
2901   <*numerical>
2902   seq.num #1 + 'seq.num :=
2903   seq.num int.to.fix
2904   </numerical>
2905   'sort.label :=
2906   sort.label *
2907   #1 entry.max$ substring$
2908   'sort.key$ :=
2909 }

```

Now comes the final computation for alphabetic labels, putting in the 'a's and 'b's and so forth if required. This involves two passes: a forward pass to put in the 'b's, 'c's and so on, and a backwards pass to put in the 'a's (we don't want to put in 'a's unless we know there are 'b's). We have to keep track of the longest (in width\$ terms) label, for use by the "thebibliography" environment.

```

VAR: longest.label, last.sort.label, next.extra: string
      longest.label.width, last.extra.num: integer

initialize.longest.label ==
BEGIN
  longest.label := ""
  last.sort.label := int.to.chr$(0)
  next.extra := ""
  longest.label.width := 0
  last.extra.num := 0
END

forward.pass ==
BEGIN
  if last.sort.label = sort.label then
    last.extra.num := last.extra.num + 1
    extra.label := int.to.chr$(last.extra.num)
  else
    last.extra.num := chr.to.int$("a")
    extra.label := ""
    last.sort.label := sort.label
  fi
END

reverse.pass ==
BEGIN
  if next.extra = "b" then
    extra.label := "a"
  fi
  label := label * extra.label
  if width$(label) > longest.label.width then
    longest.label := label
    longest.label.width := width$(label)
  fi
  next.extra := extra.label
END

```

```

2911 STRINGS { longest.label last.label next.extra }
2912
2913 INTEGERS { longest.label.width last.extra.num number.label }
2914
2915 FUNCTION {initialize.longest.label}
2916 { "" 'longest.label :=
2917   #0 int.to.chr$ 'last.label :=
2918   "" 'next.extra :=
2919   #0 'longest.label.width :=

```

```

2920 #0 'last.extra.num :=
2921 #0 'number.label :=
2922 }
2923
2924 FUNCTION {forward.pass}
2925 { last.label short.label =
2926   { last.extra.num #1 + 'last.extra.num :=
2927     last.extra.num int.to.chr$ 'extra.label :=
2928   }
2929   { "a" chr.to.int$ 'last.extra.num :=
2930     "" 'extra.label :=
2931     short.label 'last.label :=
2932   }
2933   if$
2934   number.label #1 + 'number.label :=
2935 }
2936
2937 FUNCTION {reverse.pass}
2938 { next.extra "b" =
2939   { "a" 'extra.label := }
2940   'skip$
2941   if$
2942   extra.label 'next.extra :=
2943   extra.label
2944   duplicate$ empty$
2945   'skip$
2946   { "{\natexlab{" swap$ * "}}" * }
2947   if$
2948   'extra.label :=
2949   label extra.label * 'label :=
2950 }
2951
2952 FUNCTION {bib.sort.order}
2953 { sort.label 'sort.key$ :=
2954 }
2955

```

## B.9 Write bbl file

Now we're ready to start writing the .BBL file. We begin, if necessary, with a  $\LaTeX$  macro for unnamed names in an alphabetic label; next comes stuff from the 'preamble' command in the database files. Then we give an incantation containing the command `\begin{thebibliography}{...}` where the '...' is the longest label.

We also call `init.state.consts`, for use by the output routines.

```

2956 FUNCTION {begin.bib}
2957 { preamble$ empty$
2958   'skip$
2959   { preamble$ writes$ newline$ }
2960   if$
2961   "\begin{thebibliography}{ number.label int.to.str$ * "}" *
2962   writes$ newline$
2963   terms.in.macro

```



```

2964 { "\providecommand{\biband}{和}"
2965 write$ newline$
2966 "\providecommand{\bibetal}{等}"
2967 write$ newline$
2968 }
2969 'skip$
2970 if$
2971 "\providecommand{\natexlab}[1]{#1}"
2972 write$ newline$
2973 "\providecommand{\url}[1]{#1}"
2974 write$ newline$
2975 "\expandafter\ifx\csname urlstyle\endcsname\relax\else"
2976 write$ newline$
2977 " \urlstyle{same}\fi"
2978 write$ newline$
2979 "\expandafter\ifx\csname href\endcsname\relax"
2980 write$ newline$
2981 " \DeclareUrlCommand\doi{\urlstyle{rm}}"
2982 write$ newline$
2983 " \def\epprint#1#2{#2}"
2984 write$ newline$
2985 "\else"
2986 write$ newline$
2987 " \def\doi#1{\href{https://doi.org/#1}{\nolinkurl{#1}}}"
2988 write$ newline$
2989 " \let\epprint\href"
2990 write$ newline$
2991 "\fi"
2992 write$ newline$
2993 }
2994

```

Finally, we finish up by writing the ‘\end{thebibliography}’ command.

```

2995 FUNCTION {end.bib}
2996 { newline$
2997 "\end{thebibliography}" write$ newline$
2998 }
2999

```

## B.10 Main execution

Now we read in the .BIB entries.

```

3000 READ
3001
3002 EXECUTE {init.state.consts}
3003
3004 EXECUTE {load.config}
3005
3006 ⟨*numerical⟩
3007 EXECUTE {init.seq}
3008
3009 ⟨/numerical⟩
3010 ITERATE {presort}
3011

```

And now we can sort

```
3012 SORT
3013
3014 EXECUTE {initialize.longest.label}
3015
3016 ITERATE {forward.pass}
3017
3018 REVERSE {reverse.pass}
3019
3020 ITERATE {bib.sort.order}
3021
3022 SORT
3023
3024 EXECUTE {begin.bib}
3025
```

Now we produce the output for all the entries

```
3026 ITERATE {call.type$}
3027
3028 EXECUTE {end.bib}
3029 </author-year | numerical>
```