

# The engord package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2010/03/01 v1.8

## Abstract

The package generates the suffix of English ordinal numbers. It can be used with plain and L<sup>A</sup>T<sub>E</sub>X formats.

## Contents

<b>1</b>	<b>Usage</b>	<b>2</b>
1.1	Package options . . . . .	2
1.2	Examples . . . . .	2
<b>2</b>	<b>Implementation</b>	<b>3</b>
2.1	Reload check and identification . . . . .	3
2.2	Help commands for plain compatibility . . . . .	4
2.3	User macros . . . . .	5
2.4	Suffix generation . . . . .	6
<b>3</b>	<b>Test</b>	<b>7</b>
3.1	Catcode checks for loading . . . . .	7
<b>4</b>	<b>Installation</b>	<b>9</b>
4.1	Download . . . . .	9
4.2	Bundle installation . . . . .	9
4.3	Package installation . . . . .	9
4.4	Refresh file name databases . . . . .	9
4.5	Some details for the interested . . . . .	10
<b>5</b>	<b>History</b>	<b>10</b>
	[2000/05/23 v1.0] . . . . .	10
	[2003/04/28 v1.1] . . . . .	10
	[2006/02/20 v1.2] . . . . .	10
	[2007/04/11 v1.3] . . . . .	10
	[2007/04/26 v1.4] . . . . .	11
	[2007/09/09 v1.5] . . . . .	11
	[2007/09/20 v1.6] . . . . .	11
	[2008/08/11 v1.7] . . . . .	11
	[2010/03/01 v1.8] . . . . .	11
<b>6</b>	<b>Index</b>	<b>11</b>

# 1 Usage

`\engord{<LATEX counter name>}`

It prints the value of the L<sup>A</sup>T<sub>E</sub>X counter as English ordinal number. It can be used in the same way as `\arabic`, `\roman`, or `\alph`. The command is not available in plain-`TEX`.

`\engordnumber{<any TEX number>}`

It prints the number as English ordinal number.

`\engordletters{#1}`

This command formats the English ordinal letters after the number. It defaults to `\textsuperscript`.

`\engorderror{#1}`

It can be redefined, if an other error handling is wanted. The argument is a negative number or zero.

`\engordraisetrue`  
`\engordraisefalse`

These commands set the switch `\ifengordraise` that is asked by the default `\engordletters` before raising the ordinal letters.

## 1.1 Package options

**normal:** `\engordraisefalse`

**raise:** `\engordraisetrue`

Default is raise.

## 1.2 Examples

- `\usepackage[normal]{engord}`  
`\engordnumber{1}` → 1st  
`\engordnumber{12}` → 12th  
`\engordnumber{123}` → 123rd  
`\engord{page}` → 1st (if page has the value of one)  
`\engordraisetrue`  
`\engordnumber{12}` → 12<sup>th</sup>

- The default output of a counter can be redefined:

```
\newcounter{mycounter}  
\renewcommand{\theengcounter}{\engord{mycounter}}
```

- Because the implementation of `\engord` and `\engordnumber` is kept expandable, these commands can be used to make command names with an appropriate definition of `\engordletters`:

```
\renewcommand*{\engordletters}[1]{#1}  
\@namedef{My\engordnumber{3}Command}{...}
```

This generates the command name ‘\My4rdCommand’. Since version 1.2 the redefinition can be dropped if the letters are not raised.

- If the letters should not be raised, use L<sup>A</sup>T<sub>E</sub>X package option `normal` or use

```
\engordraisefalse
```

Also `\engordletters` could be redefined for this purpose:

```
\renewcommand*{\engordletters}[1]{#1}
```

## 2 Implementation

### 2.1 Reload check and identification

```
1 \<package>
```

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```
2 \begingroup
3   \catcode44 12 % ,
4   \catcode45 12 % -
5   \catcode46 12 % .
6   \catcode58 12 % :
7   \catcode64 11 % @
8   \catcode123 1 % {
9   \catcode125 2 % }
10  \expandafter\let\expandafter\x\csname ver@engord.sty\endcsname
11  \ifx\x\relax % plain-TeX, first loading
12  \else
13    \def\empty{}%
14    \ifx\x\empty % LaTeX, first loading,
15      % variable is initialized, but \ProvidesPackage not yet seen
16    \else
17      \catcode35 6 % #
18      \expandafter\ifx\csname PackageInfo\endcsname\relax
19        \def\x#1#2{%
20          \immediate\write-1{Package #1 Info: #2.}%
21        }%
22      \else
23        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24      \fi
25      \x{engord}{The package is already loaded}%
26    \aftergroup\endinput
27  \fi
28 \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31   \catcode35 6 % #
32   \catcode40 12 % (
33   \catcode41 12 % )
34   \catcode44 12 % ,
35   \catcode45 12 % -
36   \catcode46 12 % .
37   \catcode47 12 % /
38   \catcode58 12 % :
39   \catcode64 11 % @
40   \catcode91 12 % [
41   \catcode93 12 % ]
42   \catcode123 1 % {
43   \catcode125 2 % }
44   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45     \def\x#1#2#3[#4]{\endgroup
```

```

46      \immediate\write-1{Package: #3 #4}%
47      \xdef#1{#4}%
48    }%
49  \else
50    \def\x#1#2[#3]{\endgroup
51      #2[{#3}]%
52      \ifx#1\@undefined
53        \xdef#1{#3}%
54      \fi
55      \ifx#1\relax
56        \xdef#1{#3}%
57      \fi
58    }%
59  \fi
60 \expandafter\x\csname ver@engord.sty\endcsname
61 \ProvidesPackage{engord}%
62 [2010/03/01 v1.8 Provides English ordinal numbers (H0)]

```

## 2.2 Help commands for plain compatibility

```

63 \begingroup
64   \catcode123 1 % {
65   \catcode125 2 % }
66   \def\x{\endgroup
67     \expandafter\edef\csname EO@AtEnd\endcsname{%
68       \catcode35 \the\catcode35\relax
69       \catcode64 \the\catcode64\relax
70       \catcode123 \the\catcode123\relax
71       \catcode125 \the\catcode125\relax
72     }%
73   }%
74 \x
75 \catcode35 6 % #
76 \catcode64 11 % @
77 \catcode123 1 % {
78 \catcode125 2 % }
79 \def\TMP@EnsureCode#1#2{%
80   \edef\EO@AtEnd{%
81     \EO@AtEnd
82     \catcode#1 \the\catcode#1\relax
83   }%
84   \catcode#1 #2\relax
85 }
86 \TMP@EnsureCode{33}{12}% !
87 \TMP@EnsureCode{36}{3}% $
88 \TMP@EnsureCode{39}{12}% '
89 \TMP@EnsureCode{42}{12}% *
90 \TMP@EnsureCode{46}{12}% .
91 \TMP@EnsureCode{47}{12}% /
92 \TMP@EnsureCode{60}{12}% <
93 \TMP@EnsureCode{94}{7}% ^ (superscript)
94 \TMP@EnsureCode{96}{12}% '

```

\EO@def Definitions, \newcommand does not exist in plain-TeX.

```

95 \begingroup\expandafter\expandafter\expandafter\endgroup
96 \expandafter\ifx\csname newcommand\endcsname\relax
97   \def\EO@def{\def}%
98 \else
99   \def\EO@def#1{%
100     \newcommand*{#1}{}%
101     \def#1%
102   }%
103 \fi

```

```

104 \begingroup\expandafter\expandafter\expandafter\endgroup
105 \expandafter\ifx\csname RequirePackage\endcsname\relax
106   \input infwarerr.sty\relax
107   \input ltxcmds.sty\relax
108 \else
109   \RequirePackage{infwarerr}[2007/09/09]%
110   \RequirePackage{ltxcmds}[2010/03/01]%
111 \fi

```

## 2.3 User macros

**\ifengordraise** The switch `\ifengordraise`, whether the ordinal letters are raised or not. Default is raised because of compatibility.

```

112 \ltx@newif\ifengordraise
113 \engordraisetrue

```

In L<sup>A</sup>T<sub>E</sub>X this also can be controlled by option `normal` or `raise`.

```

114 \begingroup\expandafter\expandafter\expandafter\endgroup
115 \expandafter\ifx\csname DeclareOption\endcsname\relax
116 \else
117   \DeclareOption{normal}{\engordraisefalse}%
118   \DeclareOption{raise}{\engordraisetrue}%
119   \ProcessOptions*\relax
120 \fi

```

**\engordletters** `\engordletters` is called with one argument, the english ordinal letters, and contains the code to format them. It defaults to `\textsuperscript` depending on `\ifengordraise`.

```

121 \expandafter\ifx\csname engordletters\endcsname\relax
122   \EO@def\engordletters{%
123     \ifengordraise
124       \expandafter\engordtextsuperscript
125     \fi
126   }%
127 \fi

```

**\engordtextsuperscript** For plain- $\text{T}_{\text{E}}\text{X}$  the definition is quite ugly, redefine `\engordtextsuperscript` if you have a better one.

```

128 \expandafter\ifx\csname engordtextsuperscript\endcsname\relax
129   \begingroup\expandafter\expandafter\expandafter\endgroup
130   \expandafter\ifx\csname textsuperscript\endcsname\relax
131     \def\engordtextsuperscript#1{%
132       \relax
133       \ifmmode
134         ^{\rm#1}%
135       \else
136         $^{\rm#1}$%
137       \fi
138     }%
139   \else
140     \def\engordtextsuperscript{\textsuperscript}%
141   \fi
142 \fi

```

**\engorderror** `\engorderror` is called, if the number is zero or negative.

```

143 \expandafter\ifx\csname engorderror\endcsname\relax
144   \EO@def\engorderror#1{%
145     #1\engordletters{!ERROR!}%
146     \@PackageWarning{engord}{%
147       ‘#1’ is not an ordinal number%
148     }%

```

```

149 }%
150 \fi

```

**\engord** **\engord** expects a L<sup>A</sup>T<sub>E</sub>X counter name as argument and calls **\engordnumber**. It is defined only, if L<sup>A</sup>T<sub>E</sub>X is used.

```

151 \begingroup\expandafter\expandafter\expandafter\endgroup
152 \expandafter\ifx\csname newcounter\endcsname\relax
153 \else
154   \EO@def\engord#1{%
155     \engordnumber{\value{#1}}%
156   }%
157 \fi

```

**\engordnumber** **\engordnumber** is the user command to print a number as english ordinal number. The argument can be any T<sub>E</sub>X number like explicit numbers, register values, ...

In a safe way it converts the T<sub>E</sub>X number argument into a form that only consists of decimal digits.

```

158 \EO@def\engordnumber#1{%
159   \expandafter\EO@number\expandafter{\number#1}%
160 }

```

## 2.4 Suffix generation

**\EO@number** **\EO@number** expects a number with decimal digits as argument and looks at the size of the number and the count of the digits:

```

161 \def\EO@number#1{%
162   \ifnum#1<1 % handle the error case
163     \engorderror{#1}%
164   \else
165     \ifnum#1<21 %
166       \EO@ord{#1}%
167     \else
168       \ifnum#1<100 %
169         \EO@twodigits#1%
170       \else
171         \@ReturnAfterFi{%
172           \EO@reverse#1\@nil}\EO@afterreverse
173         }%
174       \fi
175     \fi
176   \fi
177 }

```

**\@ReturnAfterFi** An internal help macro to prevent a too deep **\if** nesting.

```

178 \long\def\@ReturnAfterFi#1\fi{\fi#1}

```

**\EO@ord** **\EO@ord** prints the number with ord letters.  
**#1**: decimal digits, **#1** < 21

```

179 \def\EO@ord#1{%
180   #1%
181   \expandafter\engordletters
182   \ifcase#1{th}\or
183     {st}\or
184     {nd}\or
185     {rd}\else
186     {th}%
187   \fi
188 }

```

**\EO@twodigits** **\EO@twodigits** expects a number with two digits,  
 20 < number < 100

```

189 \def\E0@twodigits#1#2{%
190   #1\E0@ord{#2}%
191 }

\E0@reverse \E0@reverse reverses the digits of the number.
#1: next digit
#2: rest of the digits
#3: already reversed digits
#4: next command to call with the reversed number as argument
192 \def\E0@reverse#1#2\@nil#3#4{%
193   \ifx\#2\%
194     #4{#1#3}%
195   \else
196     \@ReturnAfterFi{%
197       \E0@reverse#2\@nil{#1#3}{#4}%
198     }%
199   \fi
200 }

\E0@afterreverse \E0@afterreverse calls \E0@reverseback so that \E0@reverseback can inspect
the digits of the number.
201 \def\E0@afterreverse#1{%
202   \E0@reverseback#1\@nil
203 }

\E0@reverseback \E0@reverseback reverses the reversion.
#1: the last digit of the number
#2: the second last digit of the number
#3: first digits of the number in reversed order, it is not empty, because
\E0@reverseback is only called with numbers > 100.
204 \def\E0@reverseback#1#2#3\@nil{%
205   \E0@reverse#3\@nil{}\@firstofone
206   \ifnum#2#1<21 %
207     \E0@ord{#2#1}%
208   \else
209     #2\E0@ord{#1}%
210   \fi
211 }

212 \E0@AtEnd
213 \</package>

```

## 3 Test

### 3.1 Catcode checks for loading

```

214 \<test1>
215 \catcode'\{=1 %
216 \catcode'\}=2 %
217 \catcode'\#=6 %
218 \catcode'\@=11 %
219 \expandafter\ifx\csname count@\endcsname\relax
220   \countdef\count@=255 %
221 \fi
222 \expandafter\ifx\csname @gobble\endcsname\relax
223   \long\def\@gobble#1{%
224   \fi
225 \expandafter\ifx\csname @firstofone\endcsname\relax
226   \long\def\@firstofone#1{#1}%
227 \fi

```

```

228 \expandafter\ifx\csname loop\endcsname\relax
229 \expandafter\@firstofone
230 \else
231 \expandafter\@gobble
232 \fi
233 {%
234 \def\loop#1\repeat{%
235 \def\body{#1}%
236 \iterate
237 }%
238 \def\iterate{%
239 \body
240 \let\next\iterate
241 \else
242 \let\next\relax
243 \fi
244 \next
245 }%
246 \let\repeat=\fi
247 }%
248 \def\RestoreCatcodes{
249 \count@=0 %
250 \loop
251 \edef\RestoreCatcodes{%
252 \RestoreCatcodes
253 \catcode\the\count@=\the\catcode\count@\relax
254 }%
255 \ifnum\count@<255 %
256 \advance\count@ 1 %
257 \repeat
258
259 \def\RangeCatcodeInvalid#1#2{%
260 \count@=#1\relax
261 \loop
262 \catcode\count@=15 %
263 \ifnum\count@<#2\relax
264 \advance\count@ 1 %
265 \repeat
266 }
267 \expandafter\ifx\csname LoadCommand\endcsname\relax
268 \def\LoadCommand{\input engord.sty\relax}%
269 \fi
270 \def\Test{%
271 \RangeCatcodeInvalid{0}{47}%
272 \RangeCatcodeInvalid{58}{64}%
273 \RangeCatcodeInvalid{91}{96}%
274 \RangeCatcodeInvalid{123}{255}%
275 \catcode'\@=12 %
276 \catcode'\=0 %
277 \catcode'\{=1 %
278 \catcode'\}=2 %
279 \catcode'\#=6 %
280 \catcode'\[=12 %
281 \catcode'\]=12 %
282 \catcode'\%=14 %
283 \catcode'\ =10 %
284 \catcode13=5 %
285 \LoadCommand
286 \RestoreCatcodes
287 }
288 \Test
289 \csname @@end\endcsname

```



```
290 \end
291 </test1>
```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

[CTAN:macros/latex/contrib/oberdiek/engord.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/engord.pdf](#) Documentation.

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

TDS refers to the standard “A Directory Structure for T<sub>E</sub>X Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

### 4.2 Bundle installation

**Unpacking.** Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

**Script installation.** Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

### 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting docstrip archive. The files are extracted by running the `.dtx` through plain-T<sub>E</sub>X:

```
tex engord.dtx
```

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

<code>engord.sty</code>	→ <code>tex/generic/oberdiek/engord.sty</code>
<code>engord.pdf</code>	→ <code>doc/latex/oberdiek/engord.pdf</code>
<code>test/engord-test1.tex</code>	→ <code>doc/latex/oberdiek/test/engord-test1.tex</code>
<code>engord.dtx</code>	→ <code>source/latex/oberdiek/engord.dtx</code>

If you have a `docstrip.cfg` that configures and enables docstrip’s TDS installing feature, then some files can already be in the right place, see the documentation of docstrip.

### 4.4 Refresh file name databases

If your T<sub>E</sub>X distribution (teT<sub>E</sub>X, miK<sub>T</sub><sub>E</sub>X, ...) relies on file name databases, you must refresh these. For example, teT<sub>E</sub>X users run `texhash` or `mktextlsr`.

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>

## 4.5 Some details for the interested

**Attached source.** The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk engord.pdf unpack_files output .
```

**Unpacking with L<sup>A</sup>T<sub>E</sub>X.** The `.dtx` chooses its action depending on the format:

**plain-T<sub>E</sub>X:** Run `docstrip` and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for `docstrip` (really, `docstrip` does not need L<sup>A</sup>T<sub>E</sub>X), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{engord.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

**Generating the documentation.** You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfL<sup>A</sup>T<sub>E</sub>X:

```
pdflatex engord.dtx
makeindex -s gind.ist engord.idx
pdflatex engord.dtx
makeindex -s gind.ist engord.idx
pdflatex engord.dtx
```

## 5 History

[2000/05/23 v1.0]

- First public release, published in newsgroup `de.comp.text.tex`:  
“`Re: Ordinalzahlen in LaTeX?`”<sup>2</sup>

[2003/04/28 v1.1]

- Bug fix for 30, 40, 50, ..., 100, 130, ...
- `\ordletters` renamed to documented `\engordletters`.

[2006/02/20 v1.2]

- Support for plain-T<sub>E</sub>X.
- Switch `\ifengordraise` added.
- Package options `raise` and `normal` added.
- DTX framework.

[2007/04/11 v1.3]

- Line ends sanitized.

---

<sup>2</sup>Url: <http://groups.google.com/group/de.comp.text.tex/msg/738e2cb4c51759d6>

[2007/04/26 v1.4]

- Use of package infwarerr.

[2007/09/09 v1.5]

- Catcode section added.

[2007/09/20 v1.6]

- Short description fixed (George White).

[2008/08/11 v1.7]

- Code is not changed.
- URLs updated.

[2010/03/01 v1.8]

- Compatibility with ini- $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ .

## 6 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols			
$\backslash\#$ .....	217, 279	$\backslash\mathrm{count@}$ .....	220, 249, 253, 255, 256, 260, 262, 263, 264
$\backslash\%$ .....	282	$\backslash\mathrm{countdef}$ .....	220
$\backslash\@$ .....	218, 275	$\backslash\mathrm{csname}$ .....	10, 18, 44, 60, 67, 96, 105, 115, 121, 128, 130, 143, 152, 219, 222, 225, 228, 267, 289
$\backslash\@PackageWarning$ .....	146	<b>D</b>	
$\backslash\@ReturnAfterFi$ .....	171, <u>178</u> , 196	$\backslash\mathrm{DeclareOption}$ .....	117, 118
$\backslash\@firstofone$ .....	205, 226, 229	<b>E</b>	
$\backslash\@gobble$ .....	223, 231	$\backslash\mathrm{empty}$ .....	13, 14
$\backslash\@nil$ .....	172, 192, 197, 202, 204, 205	$\backslash\mathrm{end}$ .....	290
$\backslash\@undefined$ .....	52	$\backslash\mathrm{endcsname}$ .....	10, 18, 44, 60, 67, 96, 105, 115, 121, 128, 130, 143, 152, 219, 222, 225, 228, 267, 289
$\backslash[$ .....	280	$\backslash\mathrm{endinput}$ .....	26
$\backslash\backslash$ .....	193, 276	$\backslash\mathrm{engord}$ .....	2, <u>151</u>
$\backslash\{$ .....	215, 277	$\backslash\mathrm{engorderror}$ .....	2, <u>143</u> , 163
$\backslash\}$ .....	216, 278	$\backslash\mathrm{engordletters}$ .....	2, <u>121</u> , 145, 181
$\backslash]$ .....	281	$\backslash\mathrm{engordnumber}$ .....	2, 155, <u>158</u>
$\backslash\sqcup$ .....	283	$\backslash\mathrm{engordraisefalse}$ .....	2, 117
<b>A</b>		$\backslash\mathrm{engordraisetrue}$ .....	2, 113, 118
$\backslash\mathrm{advance}$ .....	256, 264	$\backslash\mathrm{engordtextsuperscript}$ .....	124, <u>128</u>
$\backslash\mathrm{aftergroup}$ .....	26	$\backslash\mathrm{EO@afterreverse}$ .....	172, <u>201</u>
<b>B</b>		$\backslash\mathrm{EO@AtEnd}$ .....	80, 81, 212
$\backslash\mathrm{body}$ .....	235, 239	$\backslash\mathrm{EO@def}$ .....	<u>95</u> , 122, 144, 154, 158
<b>C</b>		$\backslash\mathrm{EO@number}$ .....	159, <u>161</u>
$\backslash\mathrm{catcode}$ 3, 4, 5, 6, 7, 8, 9, 17, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 64, 65, 68, 69, 70, 71, 75, 76, 77, 78, 82, 84, 215, 216, 217, 218, 253, 262, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284		$\backslash\mathrm{EO@ord}$ .....	166, <u>179</u> , 190, 207, 209
		$\backslash\mathrm{EO@reverse}$ .....	172, <u>192</u> , 205
		$\backslash\mathrm{EO@reverseback}$ .....	202, <u>204</u>
		$\backslash\mathrm{EO@twodigits}$ .....	169, <u>189</u>

<b>I</b>		<code>\ProvidesPackage</code> ..... 15, 61	
<code>\ifcase</code> .....	182	<b>R</b>	
<code>\ifengordraise</code> .....	112, 123		
<code>\ifmmode</code> .....	133		
<code>\ifnum</code> ....	162, 165, 168, 206, 255, 263		
<code>\ifx</code> .....	11, 14, 18, 44, 52, 55, 96, 105, 115, 121, 128, 130, 143, 152, 193, 219, 222, 225, 228, 267		
<code>\immediate</code> .....	20, 46	<b>T</b>	
<code>\input</code> .....	106, 107, 268		
<code>\iterate</code> .....	236, 238, 240		
<b>L</b>			
<code>\LoadCommand</code> .....	268, 285		
<code>\loop</code> .....	234, 250, 261	<b>V</b>	
<code>\ltx@newif</code> .....	112		
<b>N</b>			
<code>\newcommand</code> .....	100		
<code>\next</code> .....	240, 242, 244		
<code>\number</code> .....	159	<b>W</b>	
<b>P</b>			
<code>\PackageInfo</code> .....	23	<b>X</b>	
<code>\ProcessOptions</code> .....	119		
		<code>\x</code> 10, 11, 14, 19, 23, 25, 45, 50, 60, 66, 74	