

ledmac

A presumptuous attempt to port EDMAC, TABMAC and EDSTANZA to LaTeX*

Peter Wilson
Herries Press[†]
Maïeul Rouquette[‡]

based on the original work by
John Lavagnino, Dominik Wujastyk, Herbert Breger and Wayne Sullivan

Abstract

For over ten years EDMAC, a set of PLAIN T_EX macros, has been available for typesetting critical editions in the traditional way, i.e., similar to the Oxford Classical Texts, Teubner, Arden Shakespeare and other series. A separate set of PLAIN T_EX macros, TABMAC, provides for tabular material. Another set of PLAIN T_EX macros, EDSTANZA, assists in typesetting verse.

The ledmac package makes the EDMAC, TABMAC and EDSTANZA facilities available to authors who would prefer to use LaTeX. The principal functions provided by the package are marginal line numbering and multiple series of footnotes and endnotes keyed to line numbers.

In addition to the EDMAC, TABMAC and EDSTANZA functions the package also provides for index entries keyed to both page and line numbers. Multiple series of the familiar numbered footnotes are also available.

Other LaTeX packages for critical editions include EDNOTES, and poemscol for poetical works.

To report bugs, please go to ledmac's GitHub page and click "New Issue": <https://github.com/maieul/ledmac/issues/>. You must open an account with github.com to access my page (maieul/ledmac). GitHub accounts are free for open-source users.

Contents

1	Introduction	4
1.1	Overview	5
1.2	History	6

*This file (ledmac.dtx) has version number v0.16, last revised 2012/06/21.

[†]herries dot press at earthlink dot net

[‡]maieul at maieul dot net

1.2.1	EDMAC	6
1.2.2	ledmac	8
2	The ledmac package	8
3	Numbering text lines and paragraphs	8
3.1	Lineation commands	11
3.2	Changing the line numbers	12
4	The apparatus	13
4.1	Alternate footnote formatting	16
4.2	Creating a new series	17
5	Fonts	17
6	Verse	19
7	Grouping	22
8	Crop marks	22
9	Endnotes	22
10	Cross referencing	23
11	Side notes	25
12	Familiar footnotes	25
13	Indexing	27
14	Tabular material	27
15	Miscellaneous	30
15.1	Hints	31
15.2	Known and suspected limitations	36
15.3	Use with other packages	37
15.4	Parallel typesetting	38
15.5	Notes for EDMAC users	39
16	Implementation overview	42
17	Preliminaries	42
17.1	Messages	44
18	Sectioning commands	46

19 Line counting	49
19.1 Choosing the system of lineation	49
19.2 List macros	53
19.3 Line-number counters and lists	54
19.4 Reading the line-list file	58
19.5 Commands within the line-list file	59
19.6 Writing to the line-list file	66
20 Marking text for notes	69
20.1 <code>\edtext</code> and <code>\critext</code> themselves	70
20.2 Substitute lemma	75
20.3 Substitute line numbers	75
21 Paragraph decomposition and reassembly	76
21.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	76
21.2 Processing one line	79
21.3 Line and page number computation	80
21.4 Line number printing	83
21.5 Pstart number printing in side	87
21.6 Add insertions to the vertical list	88
21.7 Penalties	89
21.8 Printing leftover notes	90
22 Footnotes	91
22.1 Fonts	91
22.2 Outer-level footnote commands	92
22.3 Normal footnote formatting	94
22.4 Standard footnote definitions	99
22.5 Paragraphed footnotes	100
22.5.1 Insertion of footnotes separator	106
22.6 Columnar footnotes	107
23 Output routine	111
24 Cross referencing	117
25 Endnotes	122
26 Side notes	125
27 Familiar footnotes	128
27.1 The A series footnotes	130
27.2 Footnote formats	131
27.2.1 Two column footnotes	134
27.2.2 Three column footnotes	135
27.2.3 Paragraphed footnotes	136
27.3 Other series footnotes	138

28 Minipages and such	140
29 Indexing	143
30 Macro as environment	145
31 Verse	149
32 Arrays and tables	152
33 The End	171
A Examples	172
A.1 Simple example	180
A.2 General example of features	181
A.3 Gascoigne	184
A.4 Shakespeare	187
A.5 Classical text edition	190
A.6 Nijmegen	196
A.7 Irish verse	200
References	205
Index	205
Change History	222

List of Figures

1	Output from <code>ledeasy.tex</code>	173
2	Output from <code>ledfeat.tex</code>	174
3	Output from <code>ledioc.tex</code>	175
4	Output from <code>ledarden.tex</code>	176
5	Output from <code>ledmixed.tex</code>	177
6	Output from <code>ledekker.tex</code>	178
7	Output from <code>ledbraonain.tex</code>	179

1 Introduction

The EDMAC macros [LW90] for typesetting critical editions of texts have been available for use with TeX for some years. Since EDMAC was introduced there has been a small but constant demand for a version of EDMAC that could be used with LaTeX. The ledmac package is an attempt to satisfy that request.

ledmac would not have been possible without the amazing work by John Lavagnino and Dominik Wujastyk, the original authors of EDMAC. I am very grateful for their encouragement and permission to use EDMAC as a base. The majority of both the

code and this manual are by these two. The tabular material is based on the TABMAC code [Bre96], by permission of its author, Herbert Breger. The verse-related code is by courtesy of Wayne Sullivan, the author of EDSTANZA [Sul92], who has kindly supplied more than his original macros.

I have altered their code and documentation as little as possible. In order to more easily show the debt that I owe, my few contributions are in the font you are now reading. I have not noted minor editorial changes such as replacing ‘TeX’ with ‘LaTeX’ or replacing ‘EDMAC’ with ‘ledmac’ or ‘package’. The original work is in the normal roman font.

There are places where I have not supplied some of the original EDMAC facilities, either because they are natively provided by LaTeX (such as font handling), or are available from other LaTeX packages (such as crop marks).

1.1 Overview

The ledmac package, together with LaTeX, provides several important facilities for formatting critical editions of texts in a traditional manner. Major features include:

- automatic stepped line numbering, by page or by section;
- sub-lineation within the main series of line numbers;
- variant readings automatically keyed to line numbers;
- caters for both prose and verse;
- multiple series of footnotes and endnotes;
- block or columnar formatting of footnotes;
- simple tabular material may be line numbered;
- indexing keyed to page and line numbers.

ledmac allows the scholar engaged in preparing a critical edition to focus attention wholly on the task of creating the critical text and evaluating the variant readings, text-critical notes and testimonia. LaTeX and ledmac will take care of the formatting and visual correlation of all the disparate types of information.

While ledmac can be used ‘out of the box’, with little or no customization, you may also go to the other extreme and view it as a collection of tools. Critical editions are amongst the most idiosyncratic of books (like their authors), so we have made ledmac deliberately bland in some ways, while also trying to document it reasonably well so that you can find out how to make it do what you want.

The original EDMAC can be used as a ‘stand alone’ processor or as part of a process. One example is its use as the formatting engine or ‘back end’ for the output of an automatic manuscript collation program. COLLATE, written by Peter Robinson, runs on the Apple Macintosh, can collate simultaneously up to a hundred manuscripts of any length, and provides facilities for the scholar to tailor the collation interactively. For further details of this and other related work, visit the EDMAC home page at <http://www.homepages.ucl.ac.uk/~ucgadkw/edmac/index.html>.

Apart from ledmac there are some other LaTeX packages for critical edition typesetting. As I am not an author, or even a prospective one, of any critical edition work

I cannot provide any opinions on what authors in this area might feel comfortable with or how well any of the packages meet their needs.

EDNOTES [Lüc03], by Uwe Lück and Christian Tapp, is another LaTeX package being developed for critical editions. Unlike ledmac which is based on EDMAC, EDNOTES takes a different (internal) approach and provides a different set of features. For example it provides additional facilities for overlapping lemmas and for handling tables. For more information there is a web site at <http://ednotes.sty.de.vu> or email to ednotes.sty@web.de.

The poemscol package [Bur01] by John Burt is designed for typesetting critical editions of collections of poems. I do not know how, or whether, poemscol and ledmac will work together.

Critical authors may find it useful to look at EDMAC, EDNOTES, ledmac, and poemscol to see which best meets their needs.

At the time of writing I know of two web sites, apart from the EDMAC home page, that have information on ledmac, and other programs.

- Jerónimo Leal pointed me to <http://www.guit.sssup.it/latex/critical.html>. This also mentions another package for critical editions called MauroTeX (<http://www.maurolico.unipi.it/mtex/mtex.htm>). These sites are both in Italian.
- Dirk-Jan Dekker maintains <http://www.djdekker.net/ledmac> which is a FAQ for typesetting critical editions and ledmac.

This manual contains a general description of how to use the LaTeX version of EDMAC, namely ledmac, (in sections 2 through 15.5); the complete source code for the package, with extensive documentation (in sections 16 through 33); a series of examples (in Appendix A); and an Index to the source code. We do not suggest that you need to read the source code for this package in order to use it; we provide this code primarily for reference, and many of our comments on it repeat material that is also found in the earlier sections. But no documentation, however thorough, can cover every question that comes up, and many can be answered quickly by consultation of the code. On a first reading, we suggest that you should skip from the general documentation in sections 2 through 15.5 to the examples in Appendix A, unless you are particularly interested in the innards of ledmac.

1.2 History

1.2.1 EDMAC

The original version of EDMAC was TEXTED.TEX, written by John Lavagnino in late 1987 and early 1988 for formatting critical editions of English plays.

John passed these macros on to Dominik Wujastyk who, in September–October 1988, added the footnote paragraphing mechanism, margin swapping and other changes to suit his own purposes, making the style more like that traditionally used for classical texts in Latin and Greek (e.g., the Oxford Classical Texts series). He also wrote some extra documentation and sent the files out to several people. This version of the macros was the first to be called EDMAC.

The present version was developed in the summer of 1990, with the intent of adding necessary features, streamlining and documenting the code, and further generalizing it to make it easily adaptable to the needs of editors in different disciplines. John did most of the general reworking and documentation, with the financial assistance of the Division of the Humanities and Social Sciences, California Institute of Technology. Dominik adapted the code to the conventions of Frank Mittelbach's `doc` option, and added some documentation, multiple-column footnotes, cross-references, and crop marks.¹ A description by John and Dominik of this version of EDMAC was published as 'An overview of EDMAC: a PLAIN T_EX format for critical editions', *TUGboat* 11 (1990), pp. 623–643.

From 1991 through 1994, the macros continued to evolve, and were tested at a number of sites. We are very grateful to all the members of the (now defunct) `edmac@mailbase.ac.uk` discussion group who helped us with smoothing out bugs and infelicities in the macros. Ron Whitney and our anonymous reviewer at the TUG were both of great help in ironing out last-minute wrinkles, while Ron made some important suggestions which may help to make future versions of EDMAC even more efficient. Wayne Sullivan, in particular, provided several important fixes and contributions, including adapting the Mittelbach/Schöpf 'New Font Selection Scheme' for use with PLAIN T_EX and EDMAC. Another project Wayne has worked on is a DVI post-processor which works with an EDMAC that has been slightly modified to output `\specials`. This combination enables you to recover to some extent the text of each line, as ASCII code, facilitating the creation of concordances, an *index verborum*, etc.

At the time of writing (1994), we are pleased to be able to say that EDMAC is being used for real-life book production of several interesting editions, such as the Latin texts of Euclid's *Elements*,² an edition of the letters of Nicolaus Copernicus,³ Simon Bredon's *Arithmetica*,⁴ a Latin translation by Plato of Tivoli of an Arabic astrolabe text,⁵ a Latin translation of part II of the Arabic *Algebra* by Abū Kāmil Shujā' b. Aslam,⁶ the Latin *Rithmachia* of Werinher von Tegernsee,⁷ a middle-Dutch romance epic on the Crusades,⁸ a seventeenth-century Hungarian politico-philosophical tract,⁹ an anonymous Latin compilation from Hungary enti-

¹This version of the macros was used to format the Sanskrit text in volume I of *Metarules of Pāṇinian Grammar* by Dominik Wujastyk (Groningen: Forsten, 1993).

²Gerhard Brey used EDMAC in the production of Hubert L. L. Busard and Menso Folkerts, *Robert of Chester's (?) Redaction of Euclid's Elements, the so-called Adelard II Version*, 2 vols., (Basel, Boston, Berlin: Birkhäuser, 1992).

³Being prepared at the German Copernicus Research Institute, Munich.

⁴Being prepared by Menso Folkerts *et al.*, at the Institut für Geschichte der Naturwissenschaften in Munich.

⁵Richard Lorch, Gerhard Brey *et al.*, at the same Institute.

⁶Richard Lorch, 'Abū Kāmil on the Pentagon and Decagon' in *Vestigia Mathematica*, ed. M. Folkerts and J. P. Hogendijk (Amsterdam, Atlanta: Rodopi, 1993).

⁷Menso Folkerts, 'Die *Rithmachia* des Werinher von Tegernsee', *ibid.*

⁸Geert H. M. Claassens, *De Middelnederlandse Kruisvaartromans*, (Amsterdam: Schipshower en Brinkman, 1993).

⁹Emil Hargittay, *Csáky István: Politica philosophiai Okoskodás-szerint való rendes életnek példája (1664–1674)* (Budapest: Argumentum Kiadó, 1992).

tled *Sermones Compilati in Studio Generali Quinqueecclesiensi in Regno Ungarie*,¹⁰ the collected letters and papers of Leibniz,¹¹ Theodosius's *Spherics*, the German *Algorismus* of Sacrobosco, the Sanskrit text of the *Kāśikāvṛtti* of Vāmana and Jayāditya,¹² and the English texts of Thomas Middleton's collected works, as well as the editions illustrated in Appendix A.

1.2.2 ledmac

Version 1.0 of TABMAC was released by Herbert Breger in October 1996. This added the capability for typesetting tabular material.

Version 0.01 of EDSTANZA was released by Wayne Sullivan in June 1992, to help a colleague with typesetting Irish verse.

In March 2003 Peter Wilson started an attempt to port EDMAC from TeX to LaTeX. The starting point was EDMAC version 3.16 as documented on 19 July 1994 (available from CTAN). In August 2003 the TABMAC functions were added; the starting point for these being version 1.0 of October 1996. The EDSTANZA (v0.01) functions were added in February 2004. Sidenotes and regular footnotes in numbered text were added in April 2004.

2 The ledmac package

ledmac is a three-pass package like LaTeX itself. Although your textual apparatus and line numbers will be printed even on the first run, it takes two more passes through LaTeX to be sure that everything gets to its right place. Any changes you make to the input file may similarly require three passes to get everything to the right place, if the changes alter the number of lines or notes. ledmac will tell you that you need to make more runs, when it notices, but it does not expend the labor to check this thoroughly. If you have problems with a line or two misnumbered at the top of a page, try running LaTeX once or twice more.

A file may mix *numbered* and *unnumbered* text. Numbered text is printed with marginal line numbers and can include footnotes and endnotes that are referenced to those line numbers: this is how you'll want to print the text that you're editing. Unnumbered text is not printed with line numbers, and you can't use ledmac's note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

3 Numbering text lines and paragraphs

`\beginnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, like:

¹⁰Being produced, as was the previous book, by Gyula Mayer in Budapest.

¹¹Leibniz, *Sämtliche Schriften und Briefe*, series I, III, VII, being edited by Dr. H. Breger, Dr. N. Gädeke and others, at the Leibniz-Archiv, Niedersächsische Landesbibliothek, Hannover. (see <http://www.nlb-hannover.de/Leibniz>)

¹²Being prepared at Poona and Lausanne Universities.


```
\beginnumbering
<text>
\endnumbering
```

The `\beginnumbering` macro resets the line number to zero, reads an auxiliary file called `<jobname>.nn` (where `<jobname>` is the name of the main input file for this job, and `nn` is 1 for the first numbered section, 2 for the second section, and so on), and then creates a new version of this auxiliary file to collect information during this run. The first instance of `\beginnumbering` also opens a file called `<jobname>.end` to receive the text of the endnotes. `\endnumbering` closes the `<jobname>.nn` file.

If the line numbering of a text is to be continuous from start to end, then the whole text will be typed between one pair of `\beginnumbering` and `\endnumbering` commands. But your text will most often contain chapter or other divisions marking sections that should be independently numbered, and these will be appropriate places to begin new numbered sections. `ledmac` has to read and store in memory a certain amount of information about the entire section when it encounters a `\beginnumbering` command, so it speeds up the processing and reduces memory use when a text is divided into a larger number of sections (at the expense of multiplying the number of external files that are generated).

`\pstart` Within a numbered section, each paragraph of numbered text must be marked
`\pend` using the `\pstart` and `\pend` commands:

```
\pstart
<paragraph of text>
\pend
```

Text that appears within a numbered section but isn't marked with `\pstart` and `\pend` will not be numbered.

The following example shows the proper section and paragraph markup, and the kind of output that would typically be generated:

<code>\beginnumbering</code>	
<code>\pstart</code>	
This is a sample paragraph, with	
lines numbered automatically.	
<code>\pend</code>	1 This is a sample paragraph
	2 with lines numbered
<code>\pstart</code>	3 automatically.
This paragraph too has its	4 This paragraph too
lines automatically numbered.	5 has its lines automatically
<code>\pend</code>	6 numbered.
The lines of this paragraph are	The lines of this paragraph
not numbered.	are not numbered.
<code>\pstart</code>	7 And here the numbering
And here the numbering begins	8 begins again.
again.	
<code>\pend</code>	
<code>\endnumbering</code>	

`\autopar` You can use `\autopar` to avoid the nuisance of this paragraph markup and still have every paragraph automatically numbered. The scope of the `\autopar` command needs to be limited by keeping it within a group, as follows:

```

\begingroup
\beginnumbering
\autopar

A paragraph of numbered text.      1 A paragraph of numbered
                                   2 text.

Another paragraph of numbered      3 Another paragraph of
text.                               4 numbered text.

\endnumbering
\endgroup

```

`\autopar` fails, however, on paragraphs that start with a `{` or with any other command that starts a new group before it generates any text. Such paragraphs need to be started explicitly, before the new group is opened, using `\indent`, `\noindent`, or `\leavevmode`, or using `\pstart` itself.¹³

`\firstlinenum` By default, `ledmac` numbers every 5th line. There are two counters, `firstlinenum`
`\linenumincrement` and `linenumincrement`, that control this behaviour; they can be changed using `\firstlinenum{<num>}` and `\linenumincrement{<num>}`. `\firstlinenum` specifies the first line that will have a printed number, and `\linenumincrement` is the difference between successive numbered lines. For example, to start printing numbers at the first line and to have every other line numbered:

```
\firstlinenum{1} \linenumincrement{2}
```

`\firstsublinenum` There are similar commands, `\firstsublinenum{<num>}` and `\sublinenumincrement{<num>}`
`\sublinenumincrement` for controlling sub-line numbering.

`\pausenumbering` `ledmac` stores a lot of information about line numbers and footnotes in memory
`\resumenumbering` as it goes through a numbered section. But at the end of such a section, it empties its memory out, so to speak. If your text has a very long numbered section it is possible that your LaTeX may reach its memory limit. There are two solutions to this. The first is to get a larger LaTeX with increased memory. The second solution is to split your long section into several smaller ones. The trouble with this is that your line numbering will start again at zero with each new section. To avoid this problem, we provide `\pausenumbering` and `\resumenumbering` which are just like `\endnumbering ... \beginnumbering`, except that they arrange for your line numbering to continue across the break. Use `\pausenumbering` only between numbered paragraphs:

¹³For a detailed study of the reasons for this restriction, see Barbara Beeton, 'Initiation rites', *TUGboat* **12** (1991), pp. 257–258.

```

\beginnumbering
\pstart
Paragraph of text.
\pend
\pausenumbering
1 Paragraph of
2 text.

\resumenumbering
\pstart
3 Another paragraph.
\pend
\endnumbering

```

We have defined these commands as two macros, in case you find it necessary to insert text between numbered sections without disturbing the line numbering. But if you are really just using these macros to save memory, you might as well say

```
\newcommand{\memorybreak}{\pausenumbering\resumenumbering}
```

and say `\memorybreak` between the relevant `\pend` and `\pstart`.

```

\numberpstarttrue
\numberpstartfalse
\thepstart

```

It's possible to insert a number at every `\pstart` command. You must use the `\numberpstarttrue` command to have it. You can stop the numbering with `\numberpstartfalse`. You can redefine the command `\thepstart` to change style. On each `\beginnumbering` the numbering restarts. With the `\sidepstartnumtrue` command, the number of `\pstart` will be printed in side. In this case, the line number will be not printed.

3.1 Lineation commands

```

\numberlinefalse
\numberlinetrue
\lineation

```

Line numbering can be disabled with `\numberlinefalse`. It can be enabled again with `\numberlinetrue`. Lines can be numbered either by page, by `pstart` or by section; you specify this using the `\lineation{<arg>}` macro, where `<arg>` is either `page`, `pstart` or `section`. You may only use this command at places where numbering is not in effect; you can't change the lineation system within a section. You can change it between sections: they don't all have to use the same lineation system. The package's standard setting is `\lineation{section}`. If the lineation is by `pstart`, the `pstart` number will be printed before the line number in the notes.

```

\linenummargin

```

The command `\linenummargin<location>` specifies the margin where the line numbers will be printed. The permissible value for `<location>` is one out of the list `left`, `right`, `inner`, or `outer`, for example `\linenummargin{inner}`. The package's default setting is `\linenummargin{left}`

to typeset the numbers in the left hand margin. You can change this whenever you're not in the middle of making a paragraph.

More precisely, the value of `\linenummargin` used is that in effect at the `\pend` of a numbered paragraph. Apart from an initial setting for `\linenummargin`, only change it after a `\pend`, whereupon it will apply to all following numbered paragraphs,

until changed again (changing it between a `\pstart` and `\pend` pair will apply the change to all the current paragraph).

`\firstlinenum` In most cases, you will not want a number printed for every single line of the
`\linenumincrement` text. Four LaTeX counters control the printing of marginal numbers and they can
`\firstsublinenum` be set by the macros `\firstlinenum{<num>}`, etc. `\firstlinenum` specifies the
`\sublinenumincrement` number of the first line in a section to number, and `\linenumincrement` is the in-
 crement between numbered lines. `\firstsublinenum` and `\sublinenumincrement`
 do the same for sub-lines. Initially, all these are set to 5 (e.g., `\firstlinenum{5}`).

`\linenumberlist` You can define `\linenumberlist` to specify a non-uniform distribution of printed
 line numbers. For example:

```
\def\linenumberlist{1,2,3,5,7,11,13,17,19,23,29}
```

to have numbers printed on prime-numbered lines only. There must be no spaces within the definition which consists of comma-separated decimal numbers. The numbers can be in any order but it is easier to read if you put them in numerical order. Either omitting the definition of `\linenumberlist` or following the vacuous definition

```
\def\linenumberlist{}
```

the standard numbering sequence is applied. The standard sequence is that specified by the combination of the `firstlinenum`, `linenumincrement`, `firstsublinenum` and `linenumincrement` counter values.

`\leftlinenum` When a marginal line number is to be printed, there are a lot of ways to
`\rightlinenum` display it. You can redefine `\leftlinenum` and `\rightlinenum` to change the
`\linenumsep` way marginal line numbers are printed in the left and right margins respectively; the initial versions print the number in font `\numlabfont` (described below) at a distance `\linenumsep` (initially set to one pica) from the text.

3.2 Changing the line numbers

Normally the line numbering starts at 1 for the first line of a section and steps up by one for each line thereafter. There are various common modifications of this system, however; the commands described here allow you to put such modifications into effect.

`\startsub` You insert the `\startsub` and `\endsub` commands in your text to turn sub-
`\endsub` lineation on and off. In plays, for example, stage directions are often numbered with sub-line numbers: as line 10.1, 10.2, 10.3, rather than as 11, 12, and 13. Titles and headings are sometimes numbered with sub-line numbers as well.

When sub-lineation is in effect, the line number counter is frozen and the sub-line counter advances instead. If one of these commands appears in the middle of a line, it doesn't take effect until the next line; in other words, a line is counted as a line or sub-line depending on what it started out as, even if that changes in the middle.

`\startlock` The `\startlock` command, used in running text, locks the line number at its
`\endlock` current value, until you say `\endlock`. It can tell for itself whether you are in a patch of line or sub-line numbering. One use for line-number locking is in printing poetry: there the line numbers should be those of verse lines rather than of printed lines, even when a verse line requires several printed lines.

`\lockdisp` When line-number locking is used, several printed lines may have the same line

number, and you have to specify whether you want the number attached to the first printed line or the last, or whether you just want the number printed by them all. (This assumes that, on the basis of the settings of the previous parameters, it is necessary to display a line number for this line.) You specify your preference using `\lockdisp{<arg>}`; its argument is a word, either `first`, `last`, or `all`. The package initially sets this as `\lockdisp{first}`.

`\setline`
`\advanceline` In some cases you may want to modify the line numbers that are automatically calculated: if you are printing only fragments of a work but want to print line numbers appropriate to a complete version, for example. The `\setline{<num>}` and `\advanceline{<num>}` commands may be used to change the current line's number (or the sub-line number, if sub-lineation is currently on). They change both the marginal line numbers and the line numbers passed to the notes. `\setline` takes one argument, the value to which you want the line number set; it must be 0 or greater. `\advanceline` takes one argument, an amount that should be added to the current line number; it may be positive or negative.

`\setlinenum` The `\setline` and `\advanceline` macros should only be used within a `\pstart... \pend` group. The `\setlinenum{<num>}` command can be used outside such a group, for example between a `\pend` and a `\pstart`. It sets the line number to `<num>`. It has no effect if used within a `\pstart... \pend` group

`\linenumberstyle`
`\sublinenumberstyle` Line numbers are normally printed as arabic numbers. You can use `\linenumberstyle{<style>}` to change the numbering style. `<style>` must be one of:

`Alph` Uppercase letters (A...Z).

`alph` Lowercase letters (a...z).

`arabic` Arabic numerals (1, 2, ...)

`Roman` Uppercase Roman numerals (I, II, ...)

`roman` Lowercase Roman numerals (i, ii, ...)

Note that with the `Alph` or `alph` styles, 'numbers' must be between 1 and 26 inclusive.

Similarly `\sublinenumberstyle{<style>}` can be used to change the numbering style of sub-line numbers, which is normally arabic numerals.

`\skipnumbering` When inserted into a numbered line the macro `\skipnumbering` causes the numbering of that particular line to be skipped; that is, the line number is unchanged and no line number will be printed.

4 The apparatus

`\edtext` Within numbered paragraphs, all footnotes and endnotes are generated by the `\edtext` macro:

`\edtext{<lemma>}{<commands>}`

The `<lemma>` argument is the lemma in the main text: `\edtext` both prints this as part of the text, and makes it available to the `<commands>` you specify to generate notes.

For example:

<code>I saw my friend \edtext{Smith}{</code> <code>\Afootnote{Jones C, D.}}</code> on Tuesday.	1 I saw my friend 2 Smith on Tuesday. <hr style="width: 100px; margin-left: 0;"/> 2 Smith] Jones C, D.
--	--

The lemma `Smith` is printed as part of this sentence in the text, and is also made available to the footnote that specifies a variant, `Jones C, D`. The footnote macro is supplied with the line number at which the lemma appears in the main text.

The `<lemma>` may contain further `\edtext` commands. Nesting makes it possible to print an explanatory note on a long passage together with notes on variants for individual words within the passage. For example:

<code>\edtext{I saw my friend</code> <code>\edtext{Smith}{\Afootnote{Jones</code> <code>C, D.}} on Tuesday.}{</code> <code>\Bfootnote{The date was</code> <code>July 16, 1954.}</code> <code>}</code>	1 I saw my friend 2 Smith on Tuesday. <hr style="width: 100px; margin-left: 0;"/> 2 Smith] Jones C, D. <hr style="width: 100px; margin-left: 0;"/> 1–2 I saw my friend Smith on Tuesday.] The date was July 16, 1954.
--	--

However, `\edtext` cannot handle overlapping but unnested notes—for example, one note covering lines 10–15, and another covering 12–18; a `\edtext` that starts in the `<lemma>` argument of another `\edtext` must end there, too. (The `\lemma` and `\linenum` commands may be used to generate overlapping notes if necessary.)

Commands used in `\edtext`'s second argument The second argument of the `\edtext` macro, `<commands>`, may contain a series of subsidiary commands that generate various kinds of notes.

<code>\Afootnote</code> <code>\Bfootnote</code> <code>\Cfootnote</code> <code>\Dfootnote</code> <code>\Efootnote</code> <code>\Aendnote</code> <code>\Bendnote</code> <code>\Cendnote</code> <code>\Dendnote</code> <code>\Eendnote</code> <code>\lemma</code>	Five separate series of footnotes are maintained; each macro taking one argument like <code>\Afootnote{<text>}</code> . When all five are used, the A notes appear in a layer just below the main text, followed by the rest in turn, down to the E notes at the bottom. These are the main macros that you will use to construct the critical apparatus of your text. The package provides five layers of notes in the belief that this will be adequate for the most demanding editions. But it is not hard to add further layers of notes should they be required. The package also maintains five separate series of endnotes. Like footnotes each macro takes a single argument like <code>\Aendnote{<text>}</code> . Normally, none of them is printed: you must use the <code>\doendnotes</code> macro described below (p. 22) to call for their output at the appropriate point in your document. Sometimes you want to change the lemma that gets passed to the notes. You can do this by using <code>\lemma{<alternative>}</code> within the second argument to <code>\edtext</code> , before the note commands. The most common use of this command is to abbreviate the lemma that's printed in the notes. For example:
--	---

<code>\edtext{I saw my friend</code>	1 I saw my friend
<code>\edtext{Smith}{\Afootnote{Jones</code>	2 Smith on Tuesday.
<code>C, D.}} on Tuesday.}</code>	<u>2 Smith]</u> Jones C, D.
<code>{\lemma{I \dots\ Tuesday.}</code>	<u>1-2 I ... Tuesday.]</u>
<code>\Bfootnote{The date was</code>	
<code>July 16, 1954.}</code>	The date was July 16, 1954.
<code>}</code>	

`\linenum` You can use `\linenum{⟨arg⟩}` to change the line numbers passed to the notes. The notes are actually given seven parameters: the page, line, and sub-line number for the start of the lemma; the same three numbers for the end of the lemma; and the font specifier for the lemma. As the argument to `\linenum`, you specify those seven parameters in that order, separated by vertical bars (the `|` character). However, you can retain the value computed by `ledmac` for any number by simply omitting it; and you can omit a sequence of vertical bars at the end of the argument. For example, `\linenum{|||23}` changes one number, the ending page number of the current lemma.

This command doesn't change the marginal line numbers in any way; it just changes the numbers passed to the footnotes. Its use comes in situations that `\edtext` has trouble dealing with for whatever reason. If you need notes for overlapping passages that aren't nested, for instance, you can use `\lemma` and `\linenum` to generate such notes despite the limitations of `\edtext`. If the `⟨lemma⟩` argument to `\edtext` is extremely long, you may run out of memory; here again you can specify a note with an abbreviated lemma using `\lemma` and `\linenum`. The numbers used in `\linenum` need not be entered manually; you can use the 'x-' symbolic cross-referencing commands below (p. 23) to compute them automatically.

Similarly, being able to manually change the lemma's font specifier in the notes might be important if you were using multiple scripts or languages. The form of the font specifier is three separate codes separated by `/` characters, giving the family, series, and shape codes as defined within NFSS.

Changing the names of these commands The commands for generating the apparatus have been given rather bland names, because editors in different fields have widely divergent notions of what sort of notes are required, where they should be printed, and what they should be called. But this doesn't mean you have to type `\Afootnote` when you'd rather say something you find more meaningful, like `\variant`. We recommend that you create a series of such aliases and use them instead of the names chosen here; all you have to do is put commands of this form at the start of your file:

```
\let\variant=\Afootnote
\let\explanatory=\Bfootnote
\let\trivial=\Aendnote
\let\testimonia=\Cfootnote
```

4.1 Alternate footnote formatting

If you just launch into ledmac using the commands outlined above, you will get a standard layout for your text and notes. You may be happy to accept this at the very beginning, while you get the hang of things, but the standard layout is not particularly pretty, and you will certainly want to modify it in due course. The package provides ways of changing the fonts and layout of your text, but these are not aimed at being totally comprehensive. They are enough to deal with simple variations from the norm, and to exemplify how you might go on to make more swingeing changes.

`\footparagraph` All footnotes will normally be formatted as a series of separate paragraphs in one column. But there are three other formats available for notes, and using these macros you can select a different format for a series of notes.

- `\footparagraph` formats all the footnotes of a series as a single paragraph (see figs. 3 and 5, pp. 175 and 177);
- `\foottwocol` formats them as separate paragraphs, but in two columns (see bottom notes in fig. 4, p. 176);
- `\footthreecol`, in three columns (see second layer of notes in fig. 2, p. 174).

Each of these macros takes one argument: a letter (between A and E) for the series of notes you want changed. So a text with three layers of notes might begin thus:

```
\footnormal{A}
\footthreecol{B}
\footparagraph{C}
```

This would make the A-notes ordinary, B-notes would be in three columns, and the bottom layer of notes would be formed into a paragraph on each page.

`\interparanoteglue` If you use paragraphed footnotes, the macro `\interparanoteglue` defines the glue appearing in between footnotes in the paragraph. It is a macro whose argument is the glue you want, and its initial setting is (see p. 104):

```
\interparanoteglue{1em plus .4em minus .4em}
```

You should set up the page layout parameters, and in particular the `\baselineskip` of the footnotes (this is done for you if you use the standard `\notefontsetup`), before you call any of these macros because their action depends on these; too much or too little space will be allotted for the notes on the page if these macros use the wrong values.¹⁴

¹⁴There is one tiny proviso about using paragraphed notes: you shouldn't force any explicit line-breaks inside such notes: do not use `\par`, `\break`, or `\penalty=-10000`. If you must have a line-break for some obscure reason, just suggest the break very strongly: `\penalty=-9999` will do the trick. Page 103 explains why this restriction is necessary.

4.2 Creating a new series

If you need more than 5 series of critical footnotes you can readily create extra series. For example to create a G series you have to put the following code into either a .sty package file, or into the preamble sandwiched between `\makeatletter` and `\makeatother` declarations.

```
\newcommand*\Gfootnote}[1]{%
  \ifnumberedpar@
    \xright@appenditem{\noexpand\Gfootnote{G}%
      {\l@d@nums}{\@tag}{#1}}\to\inserts@list
    \global\advance\insert@count by \@ne
  \else
    \vGfootnote{G}{{0|0|0|0|0|0|0|0}{#1}}%
  \fi\ignorespaces}
\newinsert\Gfootins

\newcommand*\mpGfootnote}[1]{%
  \ifnumberedpar@
    \xright@appenditem{\noexpand\mpvGfootnote{G}%
      {\l@d@nums}{\@tag}{#1}}\to\inserts@list
    \global\advance\insert@count by \@ne
  \else
    \mpvGfootnote{G}{{0|0|0|0|0|0|0|0}{#1}}%
  \fi\ignorespaces}
\newinsert\mpGfootins

\addfootins{G}
\footnormal{G}
```

5 Fonts

One of the most important features of the appearance of the notes, and indeed of your whole document, will be the fonts used. We will first describe the commands that give you control over the use of fonts in the different structural elements of the document, especially within the notes, and then in subsequent sections specify how these commands are used.

For those who are setting up for a large job, here is a list of the complete set of `ledmac` macros relating to fonts that are intended for manipulation by the user: `\endashchar`, `\fullstop`, `\notefontsetup`, `\notenumfont`, `\numlabfont`, and `\rbracket`.

`\notefontsetup` The `\notefontsetup` macro defines the standard size of the fonts for all your footnotes; `ledmac` initially defines this as:

```
\newcommand*\notefontsetup{\footnotesize}
```

`\notenumfont` The `\notenumfont` macro specifies the font used for the line numbers printed in notes. This will typically be a command like `\bfseries` that selects a distinctive style for the note numbers, but leaves the choice of a size up to `\notefontsetup`.

ledmac initially defines:

```
\newcommand{\notenumfont}{\normalfont}
```

thus using the main document font.

`\numlabfont`

Line numbers for the main text are usually printed in a smaller font in the margin. The `\numlabfont` macro is provided as a standard name for that font: it is initially defined as

```
\newcommand{\numlabfont}{\normalfont\scriptsize}
```

You might wish to use a different font if, for example, you preferred to have these line numbers printed using old-style numerals.

Here are some examples of how you might redefine some of the font macros.

```
\renewcommand*{\notefontsetup}{\small}
```

```
\renewcommand*{\notenumfont}{\sffamily}
```

These commands select `\small` fonts for the notes, and choose a sans font for the line numbers within notes.

`\endashchar`

`\fullstop`

`\rbracket`

A relatively trivial matter relates to punctuation. In your footnotes, there will sometimes be spans of line numbers like this: 12–34, or lines with sub-line numbers like this: 55.6. The en-dash and the full stop are taken from the same font as the numbers, and it all works nicely. But what if you wanted to use old-style numbers, like 12 and 34? These look nice in an edition, but when you use the fonts provided by PLAIN T_EX they are taken from a math font which does not have the en-dash or full stop in the same places as a text font. If you (or your macros) just typed `\oldstyle 12--34` or `\oldstyle 55.6` you would get ‘12”34’ and ‘55▷6’. So we define `\endashchar` and `\fullstop`, which produce an en-dash and a full stop respectively from the normal document font, whatever font you are using for the numbers. These two macros are used in the macros which format the line numbers in the margins and footnotes, instead of explicit punctuation. We also define an `\rbracket` macro for the right square bracket printed at the end of the lemma in many styles of textual notes (including ledmac’s standard style).

`\select@lemmafont`

We will briefly discuss `\select@lemmafont` here because it is important to know about it now, although it is not one of the macros you would expect to change in the course of a simple job. Hence it is ‘protected’ by having the `@`-sign in its name.

When you use the `\edtext` macro to mark a word in your text as a lemma, that word will normally be printed again in your apparatus. If the word in the text happens to be in a font such as italic or bold you would probably expect it to appear in the apparatus in the same font. This becomes an absolute necessity if the font is actually a different script, such as Arabic or Cyrillic. `\select@lemmafont` does the work of decoding ledmac’s data about the fonts used to print the lemma in the main text and calling up those fonts for printing the lemma in the note.

`\select@lemmafont` is a macro that takes one long argument—the cluster of line numbers passed to the note commands. This cluster ends with a code indicating what fonts were in use at the start of the lemma. `\select@lemmafont` selects the appropriate font for the note using that font specifier.

ledmac uses `\select@lemmafont` in a standard footnote format macro called `\normalfootfmt`. The footnote formats for each of the layers A to E are `\let` equal to `\normalfootfmt`. So all the layers of footnotes are formatted in the same way.

But it is also likely that you might want to have different fonts for just, say, the note numbers in layers A and B of your apparatus. To do this, make two copies of the `\normalfootfmt` macro (see p. 95)—or `\twocolfootfmt`, or the other appropriate macro ending in `-footfmt`, depending on what footnote format you have selected—and give these macros the names `\Afootfmt` and `\Bfootfmt`. Then, in these new macros, change the font specifications (and spacing, or whatever) to your liking.

As an example, in some texts the lemma in a footnote ends with a right bracket except where the lemma is an abbreviation (often typeset in italics). This requirement can be met as follows, assuming that the ‘A’ series footnote will be used.

First, define `\Afootfmt` as a modified version of the original `\normalfootfmt` (all the following should be enclosed in `\makeatletter` and `\makeatother` if it is in the preamble). The change is modifying `...#2\rbracket\enskip...` to read `...#2\rbracket\enskip...`, so that `\rbracket` is inside the group that includes the lemma argument.

```
\renewcommand{\Afootfmt}[3]{%
  \ledsetnormalparstuff
  {\notenumfont\printlines#1|}\strut\enspace
  {\select@lemmafont#1|#2\rbracket\enskip#3\strut\par}
```

Define an ‘abbreviation’ macro that kills the definition of `\rbracket`.

```
\newcommand*\nobrak{}
\newcommand{\abb}[1]{\textit{#1}\let\rbracket\nobrak\relax}
```

Finally, make sure that `\abb` is not expanded during the first processing of a line.

```
\newcommand{\morenoexpands}{%
  \let\abb=0%
}
```

Now code like the following can be used, and ‘lemma’ will be footnoted with a ‘]’ and ‘abbrv’ will have no ‘]’.

```
A sentence with a \edtext{lemma}{\Afootnote{ordinary}} in it.
A sentence with an \edtext{\abb{abbrv}}{\Afootnote{abbreviated}} in it.
```

6 Verse

In 1992 Wayne Sullivan¹⁵ wrote the EDSTANZA macros [Sul92] for typesetting verse in a critical edition. More specifically they were for handling poetry stanzas which use

¹⁵Department of Mathematics, University College, Dublin 4, Ireland

indentation to indicate rhyme or metre.

With Wayne Sullivan's permission the majority of this section has been taken from [Sul92]. I have made a few changes to enable his macros to be used in the LaTeX ledmac package.

`\stanza` Use `\stanza` at the start of a stanza. Each line in a stanza is ended by an ampersand (&), and the stanza itself is ended by putting `\&` at the end of the last line.

`\stanzaindentbase` Lines within a stanza may be indented. The indents are integer multiples of the length `\stanzaindentbase`, whose default value is 20pt.

`\setstanzaindents` In order to use the stanza macros, one must set the indentation values. First the value of `\stanzaindentbase` should be set, unless the default value 20pt is desired. Every stanza line indentation is a multiple of this.

To specify these multiples one invokes, for example
`\setstanzaindents{3,1,2,1,2}`.

The numerical entries must be whole numbers, 0 or greater, separated by commas without embedded spaces. The first entry gives the hanging indentation to be used if the stanza line requires more than one print line. If it is known that each stanza line will fit on a single print line, then this first entry should be 0; T_EX does less work in this case, but no harm ensues if the hanging indentation is not 0 but is never used. Enumeration is by stanza lines, not by print lines. In the above example the lines are indented one unit, two units, one unit, two units, with 3 units of hanging indentation in case a stanza line is too long to fit on one print line.

Since version 0.13, if the indentation is repeated every n verses of the stanza, you can define only the n first indentations, and say they are repeated, defining the value of the `stanzaindentsrepetition` counter at n . For example :

```
\setstanzaindents{0,1,0}
\setcounter{stanzaindentsrepetition}{2}
```

is like

```
\setstanzaindents{0,1,0,1,0,1,0,1,0,1,0}
```

If you don't use the `stanzaindentsrepetition` counter, make sure you have at least one more numerical entry in `\setstanzavalues` than the number of lines in the stanza. The macros make no restriction on the number of lines in a stanza. Stanza indentation values (and penalty values) obey T_EX's grouping conventions, so if one stanza among several has a different structure, its indentations (penalties) may be set within a group; the prior values will be restored when the group ends.

`\setstanzapenalties` When the stanzas run over several pages, often it is desirable that page breaks should arise between certain lines in the stanza, so a facility for including penalties after stanza lines is provided. If you are satisfied with the page breaks, you need not set the penalty values.

The command

```
\setstanzapenalties{1,5000,10100,5000,0}
```

results in a penalty of 5000 being placed after the first and third lines of the stanza, and a penalty of −100 after the second.

The first entry “1” is a control value. If it is zero, then no penalties are passed on to T_EX, which is the default. Values between 0 and 10000 are penalty values; values between 10001 and 20000 have 10000 subtracted and the result is given as a negative penalty. The mechanism used for indentations and penalties requires unsigned values less than 32768. No penalty is placed after the last line, so the final ,0 in then example above could be omitted. The control sequence `\endstanzaextra` can be defined to include a penalty. A penalty of 10000 will prevent a page break; such a penalty is included automatically where there is stanza hanging indentation. A penalty of −10000 (corresponding to the entry value 20000 in this context) forces a page break. Values in between act as suggestions as to the desirability of a page break at a given line. There is a subtle interaction between penalties and *glue*, so it may take some adjustment of skips and penalties to achieve the best results.

`\ampersand` If you need to print an & symbol in a stanza, use the `\ampersand` macro, not `\&` which will end the stanza.

`\endstanzaextra` The macro `\endstanzaextra`, if it is defined, is called at the end of a stanza. You could define this, for example, to add extra space between stanzas (by default there is no extra space between stanzas); if you are using the `memoir` class, it provides a length `\stanzaskip` which may come in handy.

`\startstanzahook` Similarly, if `\startstanzahook` is defined, it is called by `\stanza` at the start. This can be defined to do something.

`\flagstanza` Putting `\flagstanza[⟨len⟩]{⟨text⟩}` at the start of a line in a stanza (or elsewhere) will typeset `⟨text⟩` at a distance `⟨len⟩` before the line. The default `⟨len⟩` is `\stanzaindentbase`.

For example, to put a verse number before the first line of a stanza you could proceed along the lines:

```
\newcounter{stanzanum}
\setcounter{stanzanum}{0}
\newcommand*{\startstanzahook}{\refstepcounter{stanzanum}}
\newcommand{\numberit}{\flagstanza{\thestanzanum}}
...
\stanza
\numberit First line...&
    rest of stanza\&

\stanza
\numberit First line, second stanza...
```

`\hangingsymbol` It's possible to insert a symbol on each line of verse's hanging, as in French typography for ‘[’. To insert in `ledmac`, redefine macro `\hangingsymbol` with this code :

```
\renewcommand{\hangingsymbol}{[\,}
```

7 Grouping

In a `minipage` environment LaTeX changes `\footnote` numbering from arabic to alphabetic and puts the footnotes at the end of the `minipage`.

`minipage` You can put numbered text with critical footnotes in a `minipage` and the footnotes are set at the end of the `minipage`.

You can also put familiar footnotes (see section 12) in a `minipage` but unlike with `\footnote` the numbering scheme is unaltered.

`ledgroup` `Minipages`, of course, aren't broken across pages. Footnotes in a `ledgroup` environment are typeset at the end of the environment, as with `minipages`, but the environment includes normal page breaks. The environment makes no change to the `textwidth` so it appears as normal text; it just might be that footnotes appear in the middle of a page, with text above and below.

`ledgroupsize` The `ledgroupsize` environment is similar to `ledgroup` except that you must specify a width for the environment, as with a `minipage`.

`\begin{ledgroupsize}[\langle pos \rangle]{\langle width \rangle}`.

The required `\langle width \rangle` argument is the text width for the environment. The optional `\langle pos \rangle` argument is for positioning numbered text within the normal `textwidth`. It may be one of the characters:

l (left) numbered text is flush left with respect to the normal `textwidth`. This is the default.

c (center) numbered text is in the center of the `textwidth`.

r (right) numbered text is flush right with respect to the normal `textwidth`.

Note that normal text, footnotes, and so forth are all flush left.

`\begin{ledgroupsize}{\textwidth}` is effectively the same as `\begin{ledgroup}`

8 Crop marks

The `ledmac` package does not provide crop marks. These are available with either the `memoir` class [Wil02] or the `crop` package.

9 Endnotes

`\doendnotes` `\doendnotes{\langle letter \rangle}` closes the `.end` file that contains the text of the endnotes, if it's open, and prints one series of endnotes, as specified by a series-letter argument, e.g., `\doendnotes{A}`. `\endprint` is the macro that's called to print each note. It uses `\notenumfont`, `\select@lemfont`, and `\notefontsetup` to select fonts, just as the footnote macros do (see p. 17 above).

As endnotes may be printed at any point in the document they always start with the page number of where they were specified. The macro `\printnpnum{\langle num \rangle}` is used to print these numbers. Its default definition is:

`\newcommand*\printnpnum[1]{p.#1} }`

`\noendnotes` If you aren't going to have any endnotes, you can say `\noendnotes` in your file, before the first `\beginnumbering`, to suppress the generation of an unneeded `.end` file.

10 Cross referencing

The package provides a simple cross-referencing facility that allows you to mark places in the text with labels, and generate page and line number references to those places elsewhere using those labels.

`\edlabel` First you place a label in the text using the command `\edlabel{<lab>}`. `<lab>` can be almost anything you like, including letters, numbers, punctuation, or a combination—anything but spaces; you might say `\edlabel{toves-3}`, for example.¹⁶

`\edpageref` Elsewhere in the text, either before or after the `\edlabel`, you can refer to its location via `\edpageref{<lab>}`, or `\lineref{<lab>}`, or `\sublineref{<lab>}`.

`\lineref` These commands will produce, respectively, the page, line and sub-line on which the `\edlabel{<lab>}` command occurred.

`\sublineref`

An `\edlabel` command may appear in the main text, or in the first argument of `\edtext`, but not in the apparatus itself. But `\edpageref`, `\lineref` and `\sublineref` commands can also be used in the apparatus to refer to `\edlabel`'s in the text.

The `\edlabel` command works by writing macros to the LaTeX `.aux` file. You will need to process your document through LaTeX twice in order for the references to be resolved.

You will be warned if you say `\edlabel{foo}` and `foo` has been used as a label before. The `ref` commands will return references to the last place in the file marked with this label. You will also be warned if a reference is made to an undefined label. (This will also happen the first time you process a document after adding a new `\edlabel` command: the auxiliary file will not have been updated yet.)

If you want to refer to a word inside an `\edtext{...}{...}` command, the `\edlabel` should be defined inside the first argument, e.g.,

```
The \edtext{creature\edlabel{elephant} was quite
    unafraid}{\Afootnote{Of the mouse, that is.}}
```

`\xpageref` However, there are situations in which you'll want `ledmac` to return a number without displaying any warning messages about undefined labels or the like: if you want to use the reference in a context where LaTeX is looking for a number, such a warning will lead to a complaint that the number is missing. This is the case for references used within the argument to `\linenum`, for example. For this situation, three variants of the reference commands, with the `x` prefix, are supplied: `\xpageref`, `\xlineref`, and `\xsublineref`. They have these limitations: they will not tell you if the label is undefined, and they must be preceded in the file by

¹⁶More precisely, you should stick to characters in the T_EX categories of 'letter' and 'other'.

at least one of the four other cross-reference commands—e.g., a `\edlabel{foo}` command, even if you never refer to that label—since those commands can all do the necessary processing of the `.aux` file, and the `\x...` ones cannot.

`\xxref` The macros `\xxref` and `\edmakelabel` let you manipulate numbers and labels in ways which you may find helpful in tricky situations.

The `\xxref{<lab1>}{<lab2>}` command generates a reference to a sequence of lines, for use in the second argument of `\edtext`. It takes two arguments, both of which are labels: e.g., `\xxref{mouse}{elephant}`. It calls `\linenum` (q.v., p. 15 above) and sets the beginning page, line, and sub-line numbers to those of the place where `\edlabel{mouse}` was placed, and the ending numbers to those where `\edlabel{elephant}` occurs.

`\edmakelabel` Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired—for example, if you want to refer to a page and line number in another volume of your edition. In such cases, you can use the `\edmakelabel{<lab>}{<numbers>}` macro so that you can ‘roll your own’ label. For example, if you say ‘`\edmakelabel{elephant}{10|25|0}`’ you will create a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. It is usually best to collect your `\edmakelabel` statements near the top of your document, so that you can see them at a glance.

`\label` The normal `\label`, `\ref` and `\pageref` macros may be used within numbered text, and operate in the familiar fashion. As an example, here is one way of numbering paragraphs in numbered text, and then being able to refer to the paragraph numbers, in addition to line and page numbers.

```
\newcounter{para} \setcounter{para}{0}
\newcommand{\newpara}{%
  \refstepcounter{para}%
  \noindent\llap{\thepar. }\quad}
\newcommand{\oldpara}[1]{%
  \noindent\llap{\ref{#1}. }\quad}
```

The definitions of `\newpara` and `\oldpara` put the numbers in the left margin and the first line of the paragraph is indented. You can now write things like:

```
\linenummargin{right}
\beginnumbering
\pstart
\newpara\label{P1} A paragraph about \ldots
\pend
  In paragraph~\ref{P1} the author \ldots
\pstart
\oldpara{P1} This has the same
      \edtext{number}{\Afootnote{\ref{P1} is the paragraph, not line}}
  as the first paragraph.
\pend
\endnumbering
```


11 Side notes

The `\marginpar` command does not work in numbered text. Instead the package provides for non-floating sidenotes in either margin.

`\ledleftnote` `\ledleftnote{<text>}` will put `<text>` into the left margin level with where the command was issued. Similarly, `\ledrightnote{<text>}` puts `<text>` in the right margin.

`\ledsidenote` `\ledsidenote{<text>}` will put `<text>` into the margin specified by the current setting of `\sidenotemargin{<location>}`. The permissible value for `<location>` is one out of the list `left`, `right`, `inner`, or `outer`, for example `\sidenotemargin{outer}`. The package's default setting is `\sidenotemargin{right}`

to typeset `\ledsidenotes` in the right hand margin. This is the opposite to the default margin for line numbers. The style for a `\ledsidenote` follows that for a `\ledleftnote` or a `\ledrightnote` depending on the margin it is put in.

If two, say, `\ledleftnote`, commands are called in the same line the second `<text>` will obliterate the first. There is no problem though with having both a left and a right sidenote on the same line.

`\ledlsnotewidth` The left sidenote text is put into a box of width `\ledlsnotewidth` and the right text into a box of width `\ledrsnotewidth`. These are initially set to the value of `\marginparwidth`.

`\ledlsnotesep` The texts are put a distance `\ledlsnotesep` (or `\ledrsnotesep`) into the left (or right) margin. These lengths are initially set to the value of `\linenumsep`.

`\ledlsnotefontsetup` These macros specify how the sidenote texts are to be typeset. The initial definitions are:

```
\newcommand*{\ledlsnotefontsetup}{\raggedleft\footnotesize}% left
\newcommand*{\ledrsnotefontsetup}{\raggedright\footnotesize}% right
```

These can of course be changed to suit.

12 Familiar footnotes

The `footmisc` package [Fai03] by Robin Fairbairns has an option whereby sequential footnote marks in the text can be separated by commas^{3,4} like so. As a convenience `ledmac` provides this automatically.

`\multfootsep` `\multfootsep` is used as the separator between footnote markers. Its default definition is:

```
\providecommand*{\multfootsep}{\textsuperscript{\normalfont,}}
```

and can be changed if necessary.

`\footnoteA` As well as the standard LaTeX footnotes generated via `\footnote`, the package also provides three series of additional footnotes called `\footnoteA` through `\footnoteC`. These have the familiar marker in the text, and the marked text at the foot of the page can be formatted using any of the styles described for the critical footnotes. Note that the 'regular' footnotes have the series letter at the end of the

macro name whereas the critical footnotes have the series letter at the start of the name.

<code>\footnormalX</code> <code>\footparagraphX</code> <code>\foottwocolX</code> <code>\footthreecolX</code> <code>\thefootnoteA</code> <code>\bodyfootmarkA</code> <code>\footfootmarkA</code>	<p>Each of the <code>\foot...X</code> macros takes one argument which is the series letter (e.g., B). <code>\footnormalX</code> is the typical footnote format. With <code>\footparagraphX</code> the series is typeset a one paragraph, with <code>\foottwocolX</code> the notes are in two columns, and are in three columns with <code>\footthreecolX</code>.</p> <p>As well as using the <code>\foot...X</code> macros to specify the general footnote arrangement for a series, each series uses a set of macros for styling the marks. The mark numbering scheme is defined by the <code>\thefootnoteA</code> macro; the default is:</p>
---	--

```
\renewcommand*{\thefootnoteA}{\arabic{footnoteA}}
```

The appearance of the mark in the text is controlled by `\bodyfootmarkA` which is defined as:

```
\newcommand*{\bodyfootmarkA}{%
  \hbox{\textsuperscript{\normalfont\thefootnoteA}}}
```

The command `\footfootmarkA` controls the appearance of the mark at the start of the footnote text. It is defined as:

```
\newcommand*{\footfootmarkA}{\textsuperscript{\thefootnoteA}}
```

There are similar command triples for the other series.

Additional footnote series can be easily defined. For example, to specify a D series you have to specify the following code, either in a .sty package file or in the preamble sandwiched between `\makeatletter` and `\makeatother` commands.

```
\newcommand{\footnoteD}[1]{%
  \refstepcounter{footnoteD}%
  \@footnotemarkD
  \vfootnoteD{D}{#1}\m@mmf@prepare}
\newcounter{footnoteD}
\renewcommand{\thefootnoteD}{\arabic{footnoteD}}
\newinsert\footinsD

\newcommand{\mpfootnoteD}[1]{%
  \refstepcounter{footnoteD}%
  \@footnotemarkD
  \mpvfootnoteD{D}{#1}\m@mmf@prepare}
\newinsert\footins\mpfootinsD

\addfootinsX{D}
\footnormalX{D}
```

The above creates the D series with the default layout, and perhaps that is all that is required. If not, then you can now start to specialise it. For instance, to have the marks in the main text as lowercase roman numerals in parentheses, the marks in the foot on the baseline with a single closing parenthesis, and using the paragraph style:

```
\renewcommand*{\thefootnoteD}{\roman{footnoteD}}
\renewcommand*{\bodyfootmarkD}{\hbox{\textsuperscript{(\thefootnoteD)}}}
\renewcommand*{\footfootmarkD}{\thefootnoteD} }
\footparagraphX{D}
```

13 Indexing

`\edindex` LaTeX provides the `\index{⟨item⟩}` command for specifying that `⟨item⟩` and the current page number should be added to the raw index (`idx`) file. The `\edindex{⟨item⟩}` macro can be used in numbered text to specify that `⟨item⟩` and the current page & linenumber should be added to the raw index file.

If the memoir class is used then the macro takes an optional argument, which is the name of a raw index file. For example `\edindex[line]{item}` will use `line.idx` as the raw file instead of `\jobname.idx`.

`\pagelinesep` The page & linenumber combination is written as `page\pagelinesep line`, where the default definition is `\newcommand{\pagelinesep}{-}` so that an item on page 3, line 5 will be noted as being at 3-5. You can renew `\pagelinesep` to get a different separator (but it just so happens that `-` is the default separator used by the MAKEINDEX program).

`\edindexlab` The `\edindex` process uses a `\label/\ref` mechanism to get the correct line number. It automatically generates labels of the form `\label{\edindexlab N}`, where `N` is a number, and the default definition of `\edindexlab` is:

```
\newcommand*{\edindexlab}{\&\&}
```

in the hopes that this will not be used by any other labels (`\edindex`'s labels are like `\label{\&\&27}`). You can change `\edindexlab` to something else if you need to.

14 Tabular material

LaTeX's normal tabular and array environments cannot be used where line numbering is being done; more precisely, they can be used but with odd results, so don't use them. However, ledmac provides some simple tabulation environments that can be line numbered. The environments can also be used in normal unnumbered text.

`edarrayl` There are six environments; the `edarray*` environments are for math and
`edarrayc` `edtabular*` for text entries. The final `l`, `c`, or `r` in the environment names indi-
`edarrayr` cate that the entries will be flushleft (`l`), centered (`c`) or flushright (`r`). There is no
`edtabularl` means of specifying different formats for each column, nor for specifying a fixed width
`edtabularc` for a column. The environments are centered with respect to the surrounding text.
`edtabularr`

```
\begin{edtabularc}
```

```
1 & 2 & 3 \\\
```

```
a & bb & ccc \\\
```

```
AAA & BB & C
```

```
\end{edtabularc}
```

1	2	3
a	bb	ccc
AAA	BB	C

Entries in the environments are the same as for the normal array and tabular environments but there must be no ending `\\` at the end of the last row. *There must be the same number of column designators (the &) in each row.* There is no equivalent to any line drawing commands (such as `\hline`). However, unlike the normal environments, the `ed...` environments can cross page breaks.

Macros like `\edtext` can be used as part of an entry.
For example:

```
\beginnumbering
\pstart
\begin{edtabularl}
\textbf{\Large I} & wish I was a little bug\edindex{bug} &
\textbf{\Large I} & eat my peas with honey\edindex{honey} \\
& With whiskers \edtext{round}{\Afootnote{around}} my tummy &
& I've done it all my life. \\
& I'd climb into a honey\edindex{honey} pot &
& It makes the peas taste funny \\
& And get my tummy gummy.\edindex{gummy} &
& But it keeps them on the knife.
\end{edtabularr}
\pend
\endnumbering
```

produces the following parallel pair of verses.

1	! wish I was a little bug	! eat my peas with honey
2	With whiskers round my tummy	I've done it all my life.
3	I'd climb into a honey pot	It makes the peas taste funny
4	And get my tummy gummy.	But it keeps them on the knife.

`\edtabcolsep` The distance between the columns is controlled by the length `\edtabcolsep`.
`\spreadmath` `\spreadmath{<math>}` typesets `{<math>}` but the `{<math>}` has no effect on the
`\spreadtext` calculation of column widths. `\spreadtext{<text>}` is the analagous command for
 use in `edtabular` environments.

<code>\begin{edarrayl}</code>	1 & 2 & 3 & 4 \\	1 & 2 & 3 & 4
	& <code>\spreadmath{F+G+C}</code> & & \\	$F + G + C$
	a & bb & ccc & dddd	$a \quad bb \quad ccc \quad dddd$
<code>\end{edarrayl}</code>		

`\edrowfill` The macro `\edrowfill{<start>}{<end>}{<fill>}` fills columns number `<start>` to
`<end>` inclusive with `<fill>`. The `<fill>` argument can be any horizontal 'fill'. For example
`\hrulefill` or `\upbracefill`.

Note that every row must have the same number of columns, even if some would
 not appear to be necessary.

The `\edrowfill` macro can be used in both `tabular` and `array` environments. The
 typeset appearance of the following code is shown below.

<code>\begin{edtabularr}</code>	
1	& 2 & & 3 & & 4 & & 5 \\
Q	& & fd & & h & & qwertziohg \\
v	& wptz & & x & & y & & vb \\
g	& nnn & & \edrowfill{3}{5}{\upbracefill} & & & & & \\
<code>\edrowfill{1}{3}{\downbracefill}</code>	& & & & pq & & dgh \\

`\edbeforetab` `\edbeforetab{<text>}{<entry>}`, where `<entry>` is an entry in the leftmost column, typesets `<text>` left justified before the `<entry>`. Similarly `\edaftertab{<entry>}{<text>}`, where `<entry>` is an entry in the rightmost column, typesets `<text>` right justified after the `<entry>`.

For example:

```
\begin{edarrayl}
      A & 1 & 2 & 3 & \\
\edbeforetab{Before}{B} & 1 & 3 & 6 & \\
      C & 1 & 4 & & \edaftertab{8}{After} \\
      D & 1 & 5 & 0 & 
\end{edarrayl}
```

	$\begin{array}{rrrr} A & 1 & 2 & 3 \\ B & 1 & 3 & 6 \\ C & 1 & 4 & 8 \\ D & 1 & 5 & 0 \end{array}$		After
Before			

`\edvertline` The macro `\edvertline{<height>}` draws a vertical line `<height>` high (contrast this with `\edatright` where the size argument is half the desired height).

```
\edvertdots
\begin{edarrayr}
a & b & C & d & & \\
v & w & x & y & & \\
m & n & o & p & & \\
k & & L & cvb & & \edvertline{4pc}
\end{edarrayr}
```

a	b	C	d	
v	w	x	y	
m	n	o	p	
k		L	cvb	

The `\edvertdots` macro is similar to `\edvertline` except that it produces a vertical dotted instead of a solid line.

15 Miscellaneous

`\extensionchars` When the package assembles the name of the auxiliary file for a section, it prefixes `\extensionchars` to the section number. This is initially defined to be empty, but you can add some characters to help distinguish these files if you like; what you use is likely to be system-dependent. If, for example, you said `\renewcommand{\extensionchars}{!}`, then you would get temporary files called `jobname.!1`, `jobname.!2`, etc.

`\ifledfinal` The package can take options. The option 'final', which is the default is for final

typesetting; this sets `\ifledfinal` to TRUE. The other option, 'draft', may be useful during earlier stages and sets `\ifledfinal` to FALSE.

`\showlemma` The lemma within the text is printed via `\showlemma{lemma}`. Normally, or with the 'final' option, the definition of `\showlemma` is:

```
\newcommand*{\showlemma}[1]{#1}
```

so it just produces its argument. With the 'draft' option it is defined as

```
\newcommand*{\showlemma}[1]{\textit{#1}}
```

so that its argument is typeset in an italic font, which may make it easier to check that all lemmas have been treated.

If you would prefer some other style, you could put something like this in the preamble:

```
\ifledfinal\else
  \renewcommand{\showlemma}[1]{\textbf{#1}}% or simply ...[1]{#1}
\fi
```

`\ledplinenumtrue` Following the declaration `\ledplinenumtrue` critical footnotes will be marked with their line number. After `\ledplinenumfalse` the footnotes will be marked by `\sympmlinenum`, whose default definition is

```
\newcommand*{\sympmlinenum}{}
```

15.1 Hints

By doing a little work it is possible, for example, to set things up so that a particular footnote series only prints the linenum for the first footnote on a line.¹⁷ You may wish to skip the following but if not read it in conjunction with the code definitions from section 22.3. Suppose that we only want this to apply to the B series of normal footnotes. To accomplish this goal we have to modify the definition of `\normalvfootnote` as follows:

```
\makeatletter
\newcommand*{\previous@B@number}{-1}
\newcommand*{\previous@page}{-1}
\renewcommand*{\normalvfootnote}[2]{
  \insert\cename #1footins\endcename\bgroup
  \notefontsetup
  \footplitskip
  \spaceskip=\z@skip \xspaceskip=\z@skip
  \l@dparsfootspec #2\ledplinenumtrue%
  \ifnum\@nameuse{\previous@#1@number} = \l@dparsedstartline\relax
    \ledplinenumfalse
  \fi
  \ifnum\previous@page=\l@dparsedstartpage\relax
  \else \ledplinenumtrue \fi
  \ifnum\l@dparsedstartline=\l@dparsedendline\relax
  \else \ledplinenumtrue \fi
```

¹⁷This was requested by Dirk-Jan Dekker (djdekker@let.ru.nl).

```

\expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsestartline}%
\xdef\previous@page{\l@dparsestartpage}%          % TO HERE
\csname #1footfmt\endcsname #2\egroup}
\footnormal{B}
\makeatother

```

The additional code uses `\l@dparsefootspec` to get the footnote's line number as `\l@dparsestartline` and the page number as `\l@dparsestartpage`. It then sets `\ledplinenum` according to whether or not `\l@dparsestartline` is the same as the previous (`\previous@B@number`) number. If the page number has changed then the line number must be printed. If the starting line number is not the same as the ending line number then the line number must be printed. After `\ledplinenum` has been set the two previous values are updated to the current line and page numbers.

After the redefinition of `\normalvfootnote` the B series has to be respecified as normal for the changes to take effect. The A series will still be in the traditional style of printing every line number. To eliminate duplicate printing from the normal A series, you simply need to define `\previous@A@number` and respecify the series.

Similar techniques can be used for the other footnote styles.

Dirk-Jan Dekker felt that there was too much empty space if the starting line number was omitted in a footnote. He proposed¹⁸ this solution, here applied to a paragraphed footnote.

```

\renewcommand*{\Bparafootfmt}[3]{%
\ledsetnormalparstuff
\scriptsize
\notenumfont\printlines#1|%          % NEW FROM HERE
\ifledplinenum
\enspace
\else
{\hskip 0em plus 0em minus .4em}%
\fi%          % TO HERE
{\select@lemmafont#1|#2}\rbracket\enskip
#3\penalty-10}

```

Another question has been how to control the printing, or not, of line numbers in the footnote from the `\edtext` command. Here is an awful hack to do this. The example is an extension of the code just above.

```

\newcounter{killnum}
\setcounter{killnum}{0}
\newcommand*{\killnumbers}{\setcounter{killnum}{-1}}
\newcommand*{\restorenumbers}{\setcounter{killnum}{0}}
\renewcommand*{\Bparafootfmt}[3]{%
\ledsetnormalparstuff
\scriptsize
\ifnum\c@killnum<\z@\ledplinenumfalse\fi%    %% NEW

```

¹⁸Posted to `comp.text.tex` on 24 January 2004.


```

\notenumfont\printlines#1|%
\ifledplinenum
  \enspace
\else
  {\hskip 0em plus 0em minus .4em}%
\fi%
{\select@lemmafont#1|#2}\rbracket\enskip
#3\penalty-10}

```

In the text it is used like:

```

...
\edtext{text}{\Bfootnote{TEXT\killnumbers}}% later B line numbers not printed
...
\edtext{textual}{\Bfootnote{TEXTUAL\restorenumbers}}% later B numbers printed
...

```

That is, `\killnumbers` and `\restorenumbers` only take effect for the next and later `\edtexts`, not the one they are in. You have to kill/restore numbers in the note *before* you want the change.

Dirk-Jan Dekker suggested¹⁹ the following `\killnumber` macro if you want to occasionally kill a number.

```
\newcommand*{\killnumber}{\linenum{-1||-1||}}
```

Then insert

```
\ifnum#2=-1 \ledplinenumfalse\fi
```

near the start of the definition of `\printlines` so it reads

```

\def\printlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
  \ifnum#2=-1 \ledplinenumfalse\fi%      %% NEW
  \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
  ...

```

It is used like this:

```
\edtext{critical}{\killnumber\Afootnote{criticism}}
```

The `\killnumber` command will kill the line number for the one note, unlike `\killnumbers` which kills numbers for subsequent notes.

Perhaps, though, you just want a footnote series with no numbers at all (and maybe no lemma either).

```

\footparagraph{A}
\makeatletter
\def\zparafootfmt#1#2#3{%
  \ledsetnormalparstuff
  \notetextfont #3\penalty-10 }
\makeatother
\let\Afootfmt=\zparafootfmt

```

¹⁹Private communication, 17 February 2004.

```
...
\beginnumbering
\edtext{}{\Afootnote{numberless and lemmaless}}
...
```

At least one user has wanted a big space between the text and footnotes but a smaller space between each series. That is, the first printed series on a page must have a big skip and all later ones a small skip. Of course, there is no telling which will be the first on any given page; on one page there might be A, C and E series and on the next D and E.

Here is the start of a solution.

```
\newskip\prefootskip % the big initial skip
\prefootskip=3.3em plus .6em minus .6em
\newif\ifskipped \skippedfalse
\renewcommand*{\normalfootstart}[1]{%
  \ifskipped
    \vskip\skip\csname #1footins\endcsname% normal skip
  \else
    \skip\prefootskip%      first note so big skip
    \skippedtrue
  \fi
  \leftskip0pt\rightskip0pt
  \csname #1\footnoterule\endcsname}
\footnormal{A}% make sure the new \normalfootstart is used
\footnormal{B}
...
```

In addition similar changes would be required for paragraphed footnotes, footnotes in minipages, and the familiar footnotes.

Another user has had a wider ranging set of requirements:

- Number paragraphs and use the number in the notes for that paragraph;
- Duplicate a paragraph number later in the document and use it for that paragraph's notes;
- In any series of notes only use the paragraph number for the first in the paragraph
- Have some series use line nummbers in the notes and in other series have neither lemmas nor line numbers in the notes.
- Perhaps eliminate all paragraph numbers in the notes.

Here is some code that enables these requirements to be met. This should be in an environment where @ is treated as a letter. First, here is a version of \ref that returns a number even if the corresponding \label has not been defined.

```
\newcommand*{\saferef}[1]{%
```

```
\expandafter\ifx\csname r@#1\endcsname\relax 0\else
\ref{#1}\fi}
```

Now for some code for the paragraph numbering. Use `\newpara` at the start of a numbered paragraph and `\oldpara{<lab>}` at the start of a ‘re-numbered’ one, where `\label{<lab>}` has been used in the original numbered one.

```
\newcounter{para}\setcounter{para}{0}
\newcounter{thispara}\setcounter{thispara}{0}
\newcommand*{\newpara}{%
  \refstepcounter{para}%
  \setcounter{thispara}{\value{para}}%
  \noindent\textbf{\thepara. }}
\newcommand{\oldpara}[1]{%
  \noindent\setcounter{thispara}{\saferef{#1}}\textbf{\saferef{#1}. }}
```

Set up the A note series for lemmas, line numbers and non-repeated paragraph numbers, assuming paragraphed notes.

```
\newif\ifparnumfoot
  \parnumfoottrue% false to eliminate paragraph numbers in notes
\newcommand*{\previous@Aparnum}{-1}
\def\printlinesA#1|#2|#3|#4|#5|#6|#7|{\begingroup
  \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
  \ifnum\previous@Aparnum=\the\c@thispara% not a new paragraph
  \else% new paragraph, print, and update the check
    \ifparnumfoot \textbf{\thethispara.}\fi
    \xdef\previous@Aparnum{\the\c@thispara}%
  \fi
  \ifledplinenum \linenumr@p{#2}\else \symlinenum\fi
  \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
  \ifl@d@dash \endashchar\fi
  \ifl@d@pnum #4\fullstop\fi
  \ifl@d@elin \linenumr@p{#5}\fi
  \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
\endgroup}
\renewcommand*{\Afootfmt}[3]{%
  \ledsetnormalparstuff
  {\notenumfont\printlinesA#1|}\enspace
  {\select@lemmafont#1|#2|\rbracket\enskip
  #3\penalty-10 }
```

Set up the B series notes for no line numbers or lemmas, just non-repeated paragraph numbers, assuming normal notes.

```
\newcommand*{\previous@Bparnum}{-1}
\def\printlinesB#1|#2|#3|#4|#5|#6|#7|{\begingroup
  \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
  \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
  \ifl@d@dash \endashchar\fi
  \ifl@d@pnum #4\fullstop\fi
  \ifl@d@elin \linenumr@p{#5}\fi
  \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
\endgroup}
```

```

\ifnum\previous@Bparnum=\the\c@thispara% not a new paragraph
\else% new paragraph, print, and update the check
  \ifparnumfoot \textbf{\thethispara.}\fi
  \xdef\previous@Aparnum{\the\c@thispara}%
\fi
\endgroup}
\renewcommand*{\Bfootfmt}[3]{%
  \ledsetnormalparstuff
  {\notenumfont\printlinesB#1|}%\enspace
  {\select@lemmafont#1|#2}%\enskip
  #3\strut\par}

```

You can use the above like:

```

...
\newpara\label{fpara} A numbered\edtext{}\{\Bfootnote{lemma-less
and linenumber-less}} \edtext{paragraph}\{\Afootnote{chunk}} ...
...
\oldpara{fpara} \edtext{Repeated}\{\Afootnote{Again}}
paragraph\edtext{}\{\Bfootnote{Just a comment}} ...
...

```

15.2 Known and suspected limitations

In general, ledmac's system for adding marginal line numbers breaks anything that makes direct use of the LaTeX insert system, which includes marginpars, footnotes and floats.

However, you can use both `\footnote` and the familiar footnote series notes in numbered text. A `\marginpar` in numbered text will throw away its contents and send a warning message to the terminal and log file, but will do no harm.

`\parshape` cannot be used within numbered text, except in a very restricted way (see p. ??).

`\ballast` LaTeX is a three-pass system, but even after a document has been processed three times, there are some tricky situations in which the page breaks decided by T_EX never settle down. At each successive run, ledmac may oscillate between two different sets of page decisions. To stop this happening, should it arise, Wayne Sullivan suggested the inclusion of the quantity `\ballast`. The amount of `\ballast` will be subtracted from the penalties which apply to the page breaks calculated on the *previous* run through T_EX, thus reinforcing these breaks. So if you find your page breaks oscillating, say

```
\setcounter{ballast}{100}
```

or some such figure, and with any luck the page breaks will settle down. Luckily, this problem doesn't crop up at all often.

The restriction on explicit line-breaking in paragraphed footnotes, mentioned in footnote 14, p. 16, and described in more detail on p. 103, really is a nuisance if that's something you need to do. There are some possible solutions, described by Michael Downes, but this area remains unsatisfactory.

LaTeX has a reputation for putting things in the wrong margin after a page break. The `ledmac` package does nothing to improve the situation — in fact it just makes it more obvious if numbered text crosses a page (or column) boundary and the numbers are meant to flip from side to side. Try and keep the numbers in the same margin all the time. Another aspect of TeX's page breaking mechanism is that when numbering lines by the page, the first few numbers after a page break may continue as though the lines were still on the previous page.

`\pageparbreak`

If you can't resist flipping the numbers or numbering by the page, then you might find that judicious use of `\pageparbreak` may help if numbering goes awry across a page (or column) break. It tries to force TeX into partitioning the current paragraph into two invisibly joined paragraphs with a page break between them. Insert the command between the last word on one page and the first word on the next page. If later you change something earlier in the document the natural page break may be in a different place, and you will have to adjust the location of `\pageparbreak` accordingly.

`\footfudgefiddle`

For paragraphed footnotes TeX has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say to 68) to increase the estimate. You have to use `\renewcommand` for this, like:

```
\renewcommand{\footfudgefiddle}{68}
```

Help, suggestions and corrections will be gratefully received.

15.3 Use with other packages

Because of `ledmac`'s complexity it may not play well with other packages. In particular `ledmac` is sensitive to commands in the arguments to the `\edtext` and `*footnote` macros (this is discussed in more detail in section 20, and in particular the discussion about `\no@expands` and `\morenoexpands`). You will have to see what works or doesn't work in your particular case.

It is possible that `ledmac` and the `hyperref` package may work together. I have not tried this combination but past experience with `hyperref` suggests that cooperation is unlikely; `hyperref` changes many LaTeX internals and `ledmac` does things that are not normally seen in LaTeX.

`\morenoexpands`

You can define the macro `\morenoexpands` to modify macros that you call within `\edtext`. Because of the way `ledmac` numbers the lines the arguments to `\edtext` can be processed more than once and in some cases a macro should only be processed once. One example is the `\colorbox` macro from the `color` package, which you might use like this:

```
... \edtext{\colorbox{mycolor}{lemma}}{\Afootnote{...\colorbox{...}}}
```

If you actually try this²⁰ you will find LaTeX whinging 'Missing { inserted', and then things start to fall apart. The trick in this case is to specify either:

²⁰Reported by Dirk-Jan Dekker in the CTT thread 'Incompatibility of "color" package' on 2003/08/28.

```
\newcommand{\morenoexpands}{\let\colorbox=0}
```

or

```
\makeatletter
\newcommand{\morenoexpands}{\let\colorbox\@secondoftwo}
\makeatother
```

(\@secondoftwo is an internal LaTeX macro that takes two arguments and throws away the first one.) The first incantation lets color show in both the main text and footnotes whereas the second one shows color in the main text but kills it in the lemma and footnotes. On the other hand if you use \textcolor instead, like

```
... \edtext{\textcolor{mycolor}{lemma}}{\Afootnote{...\textcolor{...}}}
```

there is no need to fiddle with \morenoexpands as the color will naturally be displayed in both the text and footnotes. To kill the color in the lemma and footnotes, though, you can do:

```
\makeatletter
\newcommand{\morenoexpands}{\let\textcolor\@secondoftwo}
\makeatother
```

It took me a little while to discover all this. If you run into this sort of problem you may have to spend some time experimenting before hitting on a solution.

15.4 Parallel typesetting

ledmac and the parallel package [Eck03] do not work together — they have very different ideas about footnoting — and I do not have the skills to try and get them to cooperate. If you are trying to typeset short pieces in parallel on the same page you can try using the edtabular environment.

More likely you are wanting to typeset in parallel on opposite pages (e.g., original on the left (even numbered) pages and a translation on the right (odd numbered) pages). Essentially you will have to do all the page breaking yourself. Here's some example code that might help, though.

```
\makeatletter
\providecommand{\cleartoevenpage}{% defined in the memoir class
\clearpage%
\ifodd\c@page\hbox{}\clearpage\fi}
\providecommand{\cleartooddpage}{% defined in the memoir class
\clearpage%
\ifodd\c@page\else\hbox{}\clearpage\fi}
\makeatother
\newenvironment{parallelpages}{\cleartoevenpage}{\cleartooddpage}
```

```

\newcommand{\rightpage}{\cleartooddpage}
...
\begin{parallelpages}
\leftpage{first left page text}
\rightpage{first right page text}
\leftpage{second left page text}
...
\end{parallelpages}

```

Notes:

- The `\(left|right)page` declarations are guaranteed to start a new page of the specified kind.
- You are responsible for ensuring that each text (plus any footnotes) is not more than a page long.
- I used braces above so that would be possible to do, say,
`\renewcommand{\rightpage}[1]{}`
to comment out all the texts on the righthand pages.
- However, in general it's probably not a good idea for these macros to take the text as an argument as that would prohibit the use of any verbatim text.
- You could do things like
`\renewcommand{\rightpage}{\cleartooddpage\normalfont\itshape}`
`\renewcommand{\leftpage}{\cleartoevenpage\normalfont\sffamily}`
to have different fonts for the two texts.

I realise that the above does not eliminate the need for hand massaging but it might help in other ways.

Since the above was written I have developed the `ledpar` package [Wil04] as an adjunct to `ledmac` specifically for parallel typesetting of critical texts. This also co-operates with the `babel` package for typesetting in multiple languages. An even more recent extension is the `ledarab` package [Wil05] for handling parallel arabic text in critical editions.

15.5 Notes for EDMAC users

If you have never used EDMAC, ignore this section. If you have used EDMAC and are starting on a completely new document, ignore this section. Only read this section if you are converting an original EDMAC document to use `ledmac`.

The package still provides the original `\text` command, but it is (a) deprecated, and (b) its name has been changed²¹ to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext` Within numbered paragraphs, footnotes and endnotes are generated by forms

²¹A name like `\text` is likely to be defined by other LaTeX packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.

of the `\critext` macro:

```
\critext{⟨lemma⟩}⟨commands⟩/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

<pre>I saw my friend \critext{Smith} \Afootnote{Jones C, D.}/ on Tuesday.</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D.</pre>
---	---

The lemma `Smith` is printed as part of this sentence in the text, and is also made available to the footnote that specifies a variant, `Jones C, D`. The footnote macro is supplied with the line number at which the lemma appears in the main text.

The `⟨lemma⟩` may contain further `\critext` commands. Nesting makes it possible to print an explanatory note on a long passage together with notes on variants for individual words within the passage. For example:

<pre>\critext{I saw my friend \critext{Smith}{\Afootnote{Jones C, D.}/ on Tuesday.} \Bfootnote{The date was July 16, 1954.} /</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D. 1-2 I saw my friend Smith on Tuesday.] The date was July 16, 1954.</pre>
---	--

However, `\critext` cannot handle overlapping but unnested notes—for example, one note covering lines 10–15, and another covering 12–18; a `\critext` that starts in the `⟨lemma⟩` argument of another `\critext` must end there, too. (The `\lemma` and `\linenum` commands may be used to generate overlapping notes if necessary.)

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

```
\catcode'\<=\active
\let<=\critext
```

Then you might say `<{Smith}\variant{Jones}/`. This of course destroys the ability to use `<` in any new macro definitions, so long as it remains in effect; hence it should be used with care.

Changing the character at the end of the command requires more work:

```
\catcode'\<=\active
\def\xtext#1#2>{\critext{#1}{#2}/}
\let<=\xtext
```


This allows you to say `<{Smith}\Afootnote{Jones}>`.

Aliases for `\critext` of the first kind shown here also can't be nested—that is, you can't use the alias in the text that forms the first argument to `\critext`. (See section 20 to find out why.) Aliases of the second kind may be nested without any problem.

If you really have to use `\critext` in any of the tabular or array environments, then `\edtext` must not be used in the same environment. If you use `\critext` in one of these environments then you have to issue the declaration `\usingcritext` beforehand. The declaration `\usingedtext` must be issued to revert to the default assumption that `\edtext` will be used.

16 Implementation overview

We present the `ledmac` code in roughly the order in which it's used during a run of `TEX`. The order is *exactly* that in which it's read when you load the `ledmac` package, because the same file is used to generate this manual and to generate the LaTeX package file. Most of what follows consists of macro definitions, but there are some commands that are executed immediately—especially at the start of the code. The documentation generally describes the code from the point of view of what happens when the macros are executed, though. As each macro is introduced, its name is printed in the margin.

We begin with the commands you use to start and stop line numbering in a section of text (Section 17). Next comes the machinery for writing and reading the auxiliary file for each section that helps us count lines, and for creating list macros encoding the information from that file (Section 19); this auxiliary file will be read at the start of each section, to create those list macros, and a new version of the file will be started to collect information from the body of the section.

Next are commands for marking sections of the text for footnotes (Section 20), followed by the macros that take each paragraph apart, attach the line numbers and insertions, and send the result to the vertical list (Section 21). The footnote commands (Section 22) and output routine (Section 23) finish the main part of the processing; cross-referencing (Section 24) and endnotes (Section 25) complete the story.

In what follows, macros with an `@` in their name are more internal to the workings of `ledmac` than those made up just of ordinary letters, just as in PLAIN `TEX` (see *The TeXbook*, p.344). You are meant to be able to make free with ordinary macros, but the '`@`' ones should be treated with more respect, and changed only if you are pretty sure of what you are doing.

17 Preliminaries

I'll try and use `l@d` in macro names to help avoid name clashes, but this is not a hard and fast rule. For example, if an original EDMAC macro includes `edmac` I'll simply change that to `ledmac`.

Announce the name and version of the package, which is targetted for LaTeX2e.

```

1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{ledmac}[2012/06/21 v0.16 LaTeX port of EDMAC]
4
```

In general I have made the following modifications to the original EDMAC code:

- Replace as many `\def`'s by `\newcommand`'s as possible to avoid overwriting LaTeX macros.
- Replace user-level TeX counts by LaTeX counters.
- Use the LaTeX font handling mechanisms.

- Use LaTeX messaging and file facilities.

I'm adding final/draft options which I hope may be useful.

`\ifledfinal` Use this to remember which option is used, set and execute the options with final as the default.

```

5 \newif\ifledfinal
6 \DeclareOption{final}{\ledfinaltrue}
7 \DeclareOption{draft}{\ledfinalfalse}
8 \ExecuteOptions{final}

```

Use the starred form of `\ProcessOptions` which executes options in the order listed in the source file: class options, then listed package options, so a package option can override a class option with the same name. This was suggested by Dan Luecking in the ctt thread *Class/package option processing*, on 27 February 2004.

```

9 \ProcessOptions*\relax
10
11 % \end{macrocode
12 % \end{macro}
13 %
14 % \begin{macro}{\showlemma}
15 % \verb?\showlemma?\marg{lemma} typesets the lemma text in the body.
16 % It depends on the option.
17 % \changes{v0.4}{2004/02/29}{Added \cs{showlemma}}
18 % \begin{macrocode}
19 \ifledfinal
20 \newcommand*{\showlemma}[1]{#1}
21 \else
22 \newcommand*{\showlemma}[1]{\textit{#1}}
23 \fi
24

```

`\linenumberlist` The code for the `\linenumberlist` mechanism was given to me by Wayne Sullivan on 2004/02/11.

Initialize it as `\empty`

```

25 \let\linenumberlist=\empty
26

```

`\@l@tempcnta` In imitation of L^AT_EX, we create a couple of scratch counters.

`\@l@tempcntb` LaTeX already defines `\@tempcnta` and `\@tempcntb` but I have found in the past that it can be dangerous to use these (for example one of the AMS packages did something nasty to the ccaption package's use of one of these).

```

27 \newcount\@l@tempcnta \newcount\@l@tempcntb

```

`\ifl@dmemoir` Define a flag for if the memoir class has been used.

```

28 \newif\ifl@dmemoir
29 \@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
30

```

17.1 Messages

All the messages are grouped here as macros. This saves TeX's memory when the same message is repeated and also lets them be edited easily.

`\ledmac@warning` Write a warning message. Changed to use LaTeX capabilities.
 31 `\newcommand{\ledmac@warning}[1]{\PackageWarning{ledmac}{#1}}`

`\ledmac@error` Write an error message.
 32 `\newcommand{\ledmac@error}[2]{\PackageError{ledmac}{#1}{#2}}`

`\led@err@NumberingStarted`

`\led@err@NumberingNotStarted`

`\led@err@NumberingShouldHaveStarted`

33 `\newcommand*{\led@err@NumberingStarted}{%`
 34 `\ledmac@error{Numbering has already been started}{\@ehc}}`
 35 `\newcommand*{\led@err@NumberingNotStarted}{%`
 36 `\ledmac@error{Numbering was not started}{\@ehc}}`
 37 `\newcommand*{\led@err@NumberingShouldHaveStarted}{%`
 38 `\ledmac@error{Numbering should already have been started}{\@ehc}}`

`\led@mess@NotesChanged`

39 `\newcommand*{\led@mess@NotesChanged}{%`
 40 `\typeout{ledmac reminder: }%`
 41 `\typeout{ The number of footnotes in this section`
 42 `has changed since the last run.}%`
 43 `\typeout{ You will need to run LaTeX two more times`
 44 `before the footnote placement}%`
 45 `\typeout{ and line numbering in this section are`
 46 `correct.}}`

`\led@mess@SectionContinued`

47 `\newcommand*{\led@mess@SectionContinued}[1]{%`
 48 `\message{Section #1 (continuing the previous section)}`

`\led@err@LineationInNumbered`

49 `\newcommand*{\led@err@LineationInNumbered}{%`
 50 `\ledmac@error{You can't use \string\lineation\space within`
 51 `a numbered section}{\@ehc}}`

`\led@warn@BadLineation`

`\led@warn@BadLinenummargin`

`\led@warn@BadLockdisp`

`\led@warn@BadSubblockdisp`

52 `\newcommand*{\led@warn@BadLineation}{%`
 53 `\ledmac@warning{Bad \string\lineation\space argument}}`
 54 `\newcommand*{\led@warn@BadLinenummargin}{%`
 55 `\ledmac@warning{Bad \string\linenummargin\space argument}}`
 56 `\newcommand*{\led@warn@BadLockdisp}{%`
 57 `\ledmac@warning{Bad \string\lockdisp\space argument}}`
 58 `\newcommand*{\led@warn@BadSubblockdisp}{%`
 59 `\ledmac@warning{Bad \string\subblockdisp\space argument}}`

```

\led@warn@NoLineFile
60 \newcommand*{\led@warn@NoLineFile}[1]{%
61   \ledmac@warning{Can't find line-list file #1}}

\led@warn@BadAdvancelineSubline
\led@warn@BadAdvancelineLine
62 \newcommand*{\led@warn@BadAdvancelineSubline}{%
63   \ledmac@warning{\string\advanceline\space produced a sub-line
64     number less than zero.}}
65 \newcommand*{\led@warn@BadAdvancelineLine}{%
66   \ledmac@warning{\string\advanceline\space produced a line
67     number less than zero.}}

\led@warn@BadSetline
\led@warn@BadSetlinenum
68 \newcommand*{\led@warn@BadSetline}{%
69   \ledmac@warning{Bad \string\setline\space argument}}
70 \newcommand*{\led@warn@BadSetlinenum}{%
71   \ledmac@warning{Bad \string\setlinenum\space argument}}

\led@err@PstartNotNumbered
\led@err@PstartInPstart
\led@err@PendNotNumbered
\led@err@PendNoPstart
\led@err@AutoparNotNumbered
72 \newcommand*{\led@err@PstartNotNumbered}{%
73   \ledmac@error{\string\pstart\space must be used within a
74     numbered section}{\@ehc}}
75 \newcommand*{\led@err@PstartInPstart}{%
76   \ledmac@error{\string\pstart\space encountered while another
77     \string\pstart\space was in effect}{\@ehc}}
78 \newcommand*{\led@err@PendNotNumbered}{%
79   \ledmac@error{\string\pend\space must be used within a
80     numbered section}{\@ehc}}
81 \newcommand*{\led@err@PendNoPstart}{%
82   \ledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
83 \newcommand*{\led@err@AutoparNotNumbered}{%
84   \ledmac@error{\string\autopar\space must be used within a
85     numbered section}{\@ehc}}

\led@warn@BadAction
86 \newcommand*{\led@warn@BadAction}{%
87   \ledmac@warning{Bad action code, value \next@action.}}

\led@warn@DuplicateLabel
\led@warn@RefUndefined
88 \newcommand*{\led@warn@DuplicateLabel}[1]{%
89   \ledmac@warning{Duplicate definition of label '#1' on page \the\pageno.}}
90 \newcommand*{\led@warn@RefUndefined}[1]{%
91   \ledmac@warning{Reference '#1' on page \the\pageno\space undefined.
92     Using '000'.}}

\led@warn@NoMarginpars
93 \newcommand*{\led@warn@NoMarginpars}{%
94   \ledmac@warning{You can't use \string\marginpar\space in numbered text}}

```

```

\led@warn@BadSidenotemargin
95 \newcommand*{\led@warn@BadSidenotemargin}{%
96   \ledmac@warning{Bad \string\sidenotemargin\space argument}}

\led@warn@NoIndexFile
97 \newcommand*{\led@warn@NoIndexFile}[1]{%
98   \ledmac@warning{Undefined index file #1}}

\led@err@TooManyColumns
\led@err@UnequalColumns 99 \newcommand*{\led@err@TooManyColumns}{%
\led@err@LowStartColumn 100 \ledmac@error{Too many columns}{\@ehc}}
\led@err@HighEndColumn 101 \newcommand*{\led@err@UnequalColumns}{%
\led@err@ReverseColumns 102 \ledmac@error{Number of columns is not equal to the number
103   in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
104 \newcommand*{\led@err@LowStartColumn}{%
105   \ledmac@error{Start column is too low}{\@ehc}}
106 \newcommand*{\led@err@HighEndColumn}{%
107   \ledmac@error{End column is too high}{\@ehc}}
108 \newcommand*{\led@err@ReverseColumns}{%
109   \ledmac@error{Start column is greater than end column}{\@ehc}}

```

18 Sectioning commands

\section@num You use `\beginnumbering` and `\endnumbering` to begin and end a line-numbered section of the text; the pair of commands may be used as many times as you like within one document to start and end multiple, separately line-numbered sections. LaTeX will maintain and display a ‘section number’ as a count named `\section@num` that counts how many `\beginnumbering` and `\resumenumbers` commands have appeared; it needn’t be related to the logical divisions of your text.

\extensionchars Each section will read and write an associated ‘line-list file’, containing information used to do the numbering; the file will be called `\jobname.nn`, where `nn` is the section number. However, you may direct that an extra string be added before the `nn` in that filename, in order to distinguish these temporary files from others: that string is called `\extensionchars`. Initially it’s empty, since different operating systems have greatly varying ideas about what characters are permitted in file names. So `\renewcommand{\extensionchars}{-}` gives temporary files called `jobname.-1`, `jobname.-2`, etc.

```

110 \newcount\section@num
111 \section@num=0
112 \let\extensionchars=\empty

```

\ifnumbering The `\ifnumbering` flag is set to `true` if we’re within a numbered section (that is, between `\beginnumbering` and `\endnumbering`). You can use `\ifnumbering` in your own code to check whether you’re in a numbered section, but don’t change the flag’s value.

\numberingtrue

\numberingfalse

113 \newif\ifnumbering

\ifnumberingR In preparation for the ledpar package, these are related to the ‘left’ text of parallel
 \ifl@dpairing texts (when \ifl@dpairing is TRUE). They are explained in the ledpar manual.

\l@dpairingtrue 114 \newif\ifl@dpairing

\l@dpairingfalse 115 \l@dpairingfalse

\ifpst@rtedL 116 \newif\ifpst@rtedL

\pst@rtedLtrue 117 \pst@rtedLfalse

\pst@rtedLfalse 118 \newcount\l@dnumpstartsL

\l@dnumpstartsL 119 \newif\ifledRcol

\ifledRcol The \ifnumberingR flag is set to true if we’re within a right text numbered section.

120 \newif\ifnumberingR

\beginnumbering \beginnumbering begins a section of numbered text. When it’s executed we
 \initnumbering@reg increment the section number, initialize our counters, send a message to your terminal, and call macros to start the lineation machinery and endnote files.

The initializations here are trickier than they look. \line@list@stuff will use all of the counters that are zeroed here when it assembles the line-list and other lists of information about the lineation. But it will do all of this locally and within a group, and when it’s done the lists will remain but the counters will return to zero. Those same counters will then be used as we process the text of this section, but the assignments will be made globally. These initializations actually apply to both uses, though in all other respects there should be no direct interaction between the use of these counters and variables in the two processing steps.

121 \newcommand*{\beginnumbering}{%

122 \ifnumbering

123 \led@err@NumberingStarted

124 \endnumbering

125 \fi

126 \global\numberingtrue

127 \global\advance\section@num \@ne

128 \initnumbering@reg

129 \message{Section \the\section@num }%

130 \line@list@stuff{\jobname.\extensionchars\the\section@num}%

131 \l@dend@stuff

132 \setcounter{pstart}{1}

133 \firstpstarttrue

134 }

135 \newcommand*{\initnumbering@reg}{%

136 \global\pst@rtedLfalse

137 \global\l@dnumpstartsL \z@

138 \global\absline@num \z@

139 \global\line@num \z@

140 \global\subline@num \z@

141 \global\@lock \z@

142 \global\sub@lock \z@

143 \global\sublines@false

```

144 \global\let\next@page@num=\relax
145 \global\let\sub@change=\relax}
146

```

`\endnumbering` `\endnumbering` must follow the last text for a numbered section. It takes care of notifying you when changes have been noted in the input that require running the file through again to move everything to the right place.

```

147 \def\endnumbering{%
148   \ifnumbering
149     \global\numberingfalse
150     \normal@pars
151     \ifl@dpairing
152       \global\pst@rtedLfalse
153     \else
154       \ifx\insertlines@list\empty\else
155         \global\noteschanged@true
156       \fi
157       \ifx\line@list\empty\else
158         \global\noteschanged@true
159       \fi
160     \fi
161     \ifnoteschanged@
162       \led@mess@NotesChanged
163     \fi
164   \else
165     \led@err@NumberingNotStarted
166   \fi
167   \autoparfalse}

```

`\pausenumbering` The `\pausenumbering` macro is just the same as `\endnumbering`, but with the `\resumenumbering` `\ifnumbering` flag set to true, to show that numbering continues across the gap.²²

```

168 \newcommand{\pausenumbering}{%
169   \endnumbering\global\numberingtrue}

```

The `\resumenumbering` macro is a bit more involved, but not much. It does most of the same things as `\beginnumbering`, but without resetting the various counters. Note that no check is made by `\resumenumbering` to ensure that `\pausenumbering` was actually invoked.

```

170 \newcommand*{\resumenumbering}{%
171   \ifnumbering
172     \global\pst@rtedLtrue
173     \global\advance\section@num \@ne
174     \led@mess@SectionContinued{\the\section@num}%
175     \line@list@stuff{\jobname.\extensionchars\the\section@num}%
176     \l@dend@stuff
177   \else
178     \led@err@NumberingShouldHaveStarted
179   \endnumbering

```

²²Our thanks to Wayne Sullivan, who suggested the idea behind these macros.


```

180     \beginnumbering
181     \fi}
182

```

19 Line counting

19.1 Choosing the system of lineation

Sometimes you want line numbers that start at 1 at the top of each page; sometimes you want line numbers that start at 1 at each `\pstart`; other times you want line numbers that start at 1 at the start of each section and increase regardless of page breaks. `ledmac` can do it either way, and you can switch from one to the other within one work. But you have to choose one or the other for all line numbers and line references within each section. Here we will define internal codes for these systems and the macros you use to select them.

`\ifbypstart@` The `\ifbypage@` and `\ifbypstart@` flag specify the current lineation system:

`\bypstart@true` • line-of-page : `bypstart@ = false` and `bypage@ = true`.

`\bypstart@false` • line-of-pstart : `bypstart@ = true` and `bypage@ = false`.

`\ifbypage@`

`\bypage@true` `ledmac` will use the line-of-section system unless instructed otherwise.

`\bypage@false`

```

183 \newif\ifbypage@
184 \newif\ifbypstart@

```

`\lineation` `\lineation{<word>}` is the macro you use to select the lineation system. Its argument is a string: either `page` or `section` or `pstart`.

```

185 \newcommand*{\lineation}[1]{%
186     \ifnumbering
187         \led@err@LineationInNumbered
188     \else
189         \def\@tempa{#1}\def\@tempb{page}%
190         \ifx\@tempa\@tempb
191             \global\bypage@true
192             \global\bypstart@false
193         \else
194             \def\@tempb{pstart}%
195             \ifx\@tempa\@tempb
196                 \global\bypage@false
197                 \global\bypstart@true
198             \else
199                 \def\@tempb{section}
200                 \ifx\@tempa\@tempb
201                     \global\bypage@false
202                     \global\bypstart@false
203                 \else
204                     \led@warn@BadLineation
205                 \fi

```

```

206     \fi
207   \fi
208 \fi}}

```

`\linenummargin` You call `\linenummargin{<word>}` to specify which margin you want your line numbers in; it takes one argument, a string. You can put the line numbers in the same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. (These last two options assume that even-numbered pages will be on the left-hand side of every opening in your book.) You can change this within a numbered section, but the change may not take effect just when you'd like; if it's done between paragraphs nothing surprising should happen.

The selection is recorded in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.

```

209 \newcount\line@margin
210 \newcommand*{\linenummargin}[1]{%
211   \l@getline@margin{#1}%
212   \ifnum\@l@tempcntb>\m@ne
213     \global\line@margin=\@l@tempcntb
214   \fi}}
215 \newcommand*{\l@getline@margin}[1]{%
216   \def\@tempa{#1}\def\@tempb{left}%
217   \ifx\@tempa\@tempb
218     \@l@tempcntb \z@
219   \else
220     \def\@tempb{right}%
221     \ifx\@tempa\@tempb
222       \@l@tempcntb \@ne
223     \else
224       \def\@tempb{outer}%
225       \ifx\@tempa\@tempb
226         \@l@tempcntb \tw@
227       \else
228         \def\@tempb{inner}%
229         \ifx\@tempa\@tempb
230           \@l@tempcntb \thr@@
231         \else
232           \led@warn@BadLinenummargin
233           \@l@tempcntb \m@ne
234         \fi
235       \fi
236     \fi
237   \fi}
238

```

`\c@firstlinenum` The following counters tell `ledmac` which lines should be printed with line numbers.
`\c@linenumincrement` `firstlinenum` is the number of the first line in each section that gets a number; `linenumincrement` is the difference between successive numbered lines. The initial

values of these counters produce labels on lines 5, 10, 15, etc. `linenumincrement` must be at least 1.

```
239 \newcounter{firstlinenum}
240 \setcounter{firstlinenum}{5}
241 \newcounter{linenumincrement}
242 \setcounter{linenumincrement}{5}
```

`\c@firstsublinenum` The following parameters are just like `firstlinenum` and `linenumincrement`, but
`\c@sublinenumincrement` for sub-line numbers. `sublinenumincrement` must be at least 1.

```
243 \newcounter{firstsublinenum}
244 \setcounter{firstsublinenum}{5}
245 \newcounter{sublinenumincrement}
246 \setcounter{sublinenumincrement}{5}
247
```

`\firstlinenum` These macros can be used to set the corresponding counters.
`\linenumincrement` 248 \newcommand*{\firstlinenum}[1]{\setcounter{firstlinenum}{#1}}
`\firstsublinenum` 249 \newcommand*{\linenumincrement}[1]{\setcounter{linenumincrement}{#1}}
`\sublinenumincrement` 250 \newcommand*{\firstsublinenum}[1]{\setcounter{firstsublinenum}{#1}}
251 \newcommand*{\sublinenumincrement}[1]{\setcounter{sublinenumincrement}{#1}}
252

`\lockdisp` When line locking is being used, the `\lockdisp{<word>}` macro specifies whether
`\lock@disp` a line number—if one is due to appear—should be printed on the first printed line
`\l@getlock@disp` or on the last, or by all of them. Its argument is a word, either `first`, `last`, or
`all`. Initially, it is set to `first`.

`\lock@disp` encodes the selection: 0 for first, 1 for last, 2 for all.

```
253 \newcount\lock@disp
254 \newcommand{\lockdisp}[1]{%
255 \l@getlock@disp{#1}%
256 \ifnum\@l@tempcntb>\m@ne
257 \global\lock@disp=\@l@tempcntb
258 \else
259 \led@warn@BadLockdisp
260 \fi}%
261 \newcommand*{\l@getlock@disp}[1]{
262 \def\@tempa{#1}\def\@tempb{first}%
263 \ifx\@tempa\@tempb
264 \@l@tempcntb \z@
265 \else
266 \def\@tempb{last}%
267 \ifx\@tempa\@tempb
268 \@l@tempcntb \@ne
269 \else
270 \def\@tempb{all}%
271 \ifx\@tempa\@tempb
272 \@l@tempcntb \tw@
273 \else
```

```

274      \@l@dttempcntb \m@ne
275      \fi
276      \fi
277 \fi}
278

```

`\sublockdisp` The same questions about where to print the line number apply to sub-lines, and
`\sublock@disp` these are the analogous macros for dealing with the problem.

```

279 \newcount\sublock@disp
280 \newcommand{\sublockdisp}[1]{%
281   \l@dgetlock@disp{#1}%
282   \ifnum\@l@dttempcntb>\m@ne
283     \global\sublock@disp=\@l@dttempcntb
284   \else
285     \led@warn@BadSublockdisp
286   \fi}}
287

```

`\linenumberstyle` We provide a mechanism for using different representations of the line numbers, not
`\linenumrep` just the normal arabic.
`\linenumr@p` NOTE: In v0.7 `\linenumrep` and `\sublinenumrep` replaced the internal `\linenumr@p`
`\sublinenumberstyle` and `\sublinenumr@p`.
`\sublinenumrep` `\linenumberstyle` and `\sublinenumberstyle` are user level macros for setting
`\sublinenumr@p` the number representation (`\linenumrep` and `\sublinenumrep`) for line and sub-line
 numbers.

```

288 \newcommand*{\linenumberstyle}[1]{%
289   \def\linenumrep##1{\@nameuse{##1}{##1}}}
290 \newcommand*{\sublinenumberstyle}[1]{%
291   \def\sublinenumrep##1{\@nameuse{##1}{##1}}}
292
  Initialise the number styles to arabic.
292 \linenumberstyle{arabic}
293 \let\linenumr@p\linenumrep
294 \sublinenumberstyle{arabic}
295 \let\sublinenumr@p\sublinenumrep
296

```

`\leftlinenum` `\leftlinenum` and `\rightlinenum` are the macros that are called to print
`\rightlinenum` marginal line numbers on a page, for left- and right-hand margins respectively.
`\linenumsep` They're made easy to access and change, since you may often want to change the
`\numlabfont` styling in some way. These standard versions illustrate the general sort of thing
`\ledlinenum` that will be needed; they're based on the `\leftheadline` macro in *The TeXbook*,
 p. 416.

Whatever these macros output gets printed in a box that will be put into the appropriate margin without any space between it and the line of text. You'll generally want a kern between a line number and the text, and `\linenumsep` is provided as a standard way of storing its size. Line numbers are usually printed in a smaller font, and `\numlabfont` is provided as a standard name for that font.

When called, these macros will be executed within a group, so font changes and the like will remain local.

`\ledlinenum` typesets the line (and subline) number.

The original `\numlabfont` specification is equivalent to the LaTeX `\scriptsize` for a 10pt document.

```

297 \newlength{\linenumsep}
298 \setlength{\linenumsep}{1pc}
299 \newcommand*{\numlabfont}{\normalfont\scriptsize}
300 \newcommand*{\ledlinenum}{%
301   \numlabfont\linenumrep{\line@num}%
302   \ifsublines@
303     \ifnum\subline@num>0\relax
304       \unskip\fullstop\sublinenumrep{\subline@num}%
305     \fi
306   \fi}
307 \newcommand*{\leftlinenum}{%
308   \ledlinenum
309   \kern\linenumsep}
310 \newcommand*{\rightlinenum}{%
311   \kern\linenumsep
312   \ledlinenum}
313

```

19.2 List macros

Reminder: compare these with the LaTeX list macros in case they would be suitable instead.

We will make heavy use of lists of information, which will be built up and taken apart by the following macros; they are adapted from *The TeXbook*, pp. 378–379, which discusses their use in more detail.

These macros consume a large amount of the run-time of this code. We intend to replace them in a future version, and in anticipation of doing so have defined their interface in such a way that it is not sensitive to details of the underlying code.

`\list@create` The `\list@create` macro creates a new list. In this version of `ledmac` this macro doesn't do anything beyond initializing an empty list macro, but in future versions it may do more.

```

314 \newcommand*{\list@create}[1]{\global\let#1=\empty}

```

`\list@clear` The `\list@clear` macro just initializes a list to the empty list; in this version of `ledmac` it is no different from `\list@create`.

```

315 \newcommand*{\list@clear}[1]{\global\let#1=\empty}

```

`\xright@appenditem` `\xright@appenditem` expands an item and appends it to the right end of a list macro. We want the expansion because we'll often be using this to store the current value of a counter. It creates global control sequences, like `\xdef`, and uses two temporary token-list registers, `\@toksa` and `\@toksb`.

```

316 \newtoks\@toksa \newtoks\@toksb
317 \global\@toksa={\}
318 \long\def\xright@appenditem#1\to#2{%
319   \global\@toksb=\expandafter{#2}%
320   \xdef#2{\the\@toksb\the\@toksa\expandafter{#1}}%
321   \global\@toksb={}}

```

\xleft@appenditem **\xleft@appenditem** expands an item and appends it to the left end of a list macro; it is otherwise identical to **\xright@appenditem**.

```

322 \long\def\xleft@appenditem#1\to#2{%
323   \global\@toksb=\expandafter{#2}%
324   \xdef#2{\the\@toksa\expandafter{#1}\the\@toksb}%
325   \global\@toksb={}}

```

\gl@p The **\gl@p** macro removes the leftmost item from a list and places it in a control sequence. You say **\gl@p\l\to\z** (where **\l** is the list macro, and **\z** receives the left item). **\l** is assumed nonempty: say **\ifx\l\empty** to test for an empty **\l**. The control sequences created by **\gl@p** are all global.

```

326 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
327 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
328

```

19.3 Line-number counters and lists

Footnote references using line numbers rather than symbols can't be generated in one pass, because we don't know the line numbers till we ship out the pages. It would be possible if footnotes were never keyed to more than one line; but some footnotes gloss passages that may run for several lines, and they must be tied to the first line of the passage glossed. And even one-line passages require two passes if we want line-per-page numbering rather than line-per-section numbering.

So we run LaTeX over the text several times, and each time save information about page and line numbers in a 'line-list file' to be used during the next pass. At the start of each section—whenever **\beginnumbering** is executed—the line-list file for that section is read, and the information from it is encoded into a few list macros.

We need first to define the different line numbers that are involved in these macros, and the associated counters.

\line@num The count **\line@num** stores the line number that's used in marginal line numbering and in notes: counting either from the start of the page or from the start of the section, depending on your choice for this section. This may be qualified by **\subline@num**.

```

329 \newcount\line@num

```

\subline@num The count **\subline@num** stores a sub-line number that qualifies **\line@num**. For example, line 10 might have sub-line numbers 1, 2 and 3, which might be printed as lines 10.1, 10.2, 10.3.

```

330 \newcount\subline@num

```

`\ifsublines@` We maintain an associated flag, `\ifsublines@`, to tell us whether we're within a sub-line range or not.

`\sublines@true`
`\sublines@false` You may wonder why we don't just use the value of `\subline@num` to determine this—treating anything greater than 0 as an indication that sub-lineation is on. We need a separate flag because sub-lineation can be used together with line-number locking in odd ways: several pieces of a logical line might be interrupted by pieces of sub-lineated text, and those sub-line numbers should not return to zero until the next change in the major line number. This is common in the typesetting of English Renaissance verse drama, in which stage directions are given sub-line numbers: a single line of verse may be interrupted by several stage directions.

331 `\newif\ifsublines@`

`\absline@num` The count `\absline@num` stores the absolute number of lines since the start of the section: that is, the number we've actually printed, no matter what numbers we attached to them. This value is never printed on an output page, though `\line@num` will often be equal to it. It is used internally to keep track of where notes are to appear and where new pages start: using this value rather than `\line@num` is a lot simpler, because it doesn't depend on the lineation system in use.

332 `\newcount\absline@num`

We'll be calling `\absline@num` numbers 'absolute' numbers, and `\line@num` and `\subline@num` numbers 'visible' numbers.

`\@lock` The counts `\@lock` and `\sub@lock` tell us the state of line-number and sub-line-number locking. 0 means we're not within a locked set of lines; 1 means we're at the first line in the set; 2, at some intermediate line; and 3, at the last line.

333 `\newcount\@lock`

334 `\newcount\sub@lock`

`\line@list` Now we can define the list macros that will be created from the line-list file. We will maintain the following lists:

`\insertlines@list`
`\actionlines@list`
`\actions@list`

- `\line@list`: the page and line numbers for every lemma marked by `\edtext`. There are seven pieces of information, separated by vertical bars:

1. the starting page,
2. line, and
3. sub-line numbers, followed by the
4. ending page,
5. line, and
6. sub-line numbers, and then the
7. font specifier for the lemma.

These line numbers are all visible numbers. The font specifier is a set of four codes for font encoding, family, series, and shape, separated by / characters. Thus a lemma that started on page 23, line 35 and went on until page 24, line 3 (with no sub-line numbering), and was typeset in a normal roman font

would have a line list entry like this:

```
23|35|0|24|3|0|0T1|cmr/m/n.
```

There is one item in this list for every lemma marked by `\edtext`, even if there are several notes to that lemma, or no notes at all. `\edtext` reads the data in this list, making it available for use in the text of notes.

- `\insertlines@list`: the line numbers of lines that have footnotes or other insertions. These are the absolute numbers where the corresponding lemmas begin. This list contains one entry for every footnote in the section; one lemma may contribute no footnotes or many footnotes. This list is used by `\add@inserts` within `\do@line`, to tell it where to insert notes.
- `\actionlines@list`: a list of absolute line numbers at which we are to perform special actions; these actions are specified by the `\actions@list` list defined below.
- `\actions@list`: action codes corresponding to the line numbers in `\actionlines@list`. These codes tell `ledmac` what action it's supposed to take at each of these lines. One action, the page-start action, is generated behind the scenes by `ledmac` itself; the others, for specifying sub-lineation, line-number locking, and line-number alteration, are generated only by explicit commands in your input file. The page-start and line-number-alteration actions require arguments, to specify the new values for the page or line numbers; instead of storing those arguments in another list, we have chosen the action-code values so that they can encode both the action and the argument in these cases. Action codes greater than -1000 are page-start actions, and the code value is the page number; action codes less than -5000 specify line numbers, and the code value is a transformed version of the line number; action codes between these two values specify other actions which require no argument.

Here is the full list of action codes and their meanings:

Any number greater than -1000 is a page-start action: the line number associated with it is the first line on a page, and the action number is the page number. (The cutoff of -1000 is chosen because negative page-number values are used by some macro packages; we assume that page-number values less than -1000 are not common.) Page-start action codes are added to the list by the `\page@action` macro, which is (indirectly) triggered by the workings of the `\page@start` macro; that macro should always be called in the output routine, just before the page contents are assembled. `ledmac` calls it in `\pagecontents`.

The action code -1001 specifies the start of sub-lineation: meaning that, starting with the next line, we should be advancing `\subline@num` at each start-of-line command, rather than `\line@num`.

The action code -1002 specifies the end of sub-lineation. At the next start-of-line, we should clear the sub-line counter and start advancing the line number. The action codes for starting and ending sub-lineation are added to the list by the `\sub@action` macro, as called to implement the `\startsub` and `\endsub` macros.

The action code `-1003` specifies the start of line number locking. After the number for the current line is computed, it will remain at that value through the next line that has an action code to end locking.

The action code `-1004` specifies the end of line number locking.

The action code `-1005` specifies the start of sub-line number locking. After the number for the current sub-line is computed, it will remain at that value through the next sub-line that has an action code to end locking.

The action code `-1006` specifies the end of sub-line number locking.

The four action codes for line and sub-line number locking are added to the list by the `\do@lockon` and `\do@lockoff` macros, as called to implement the `\startlock` and `\endlock` macros.

An action code of `-5000` or less sets the current visible line number (either the line number or the sub-line number, whichever is currently being advanced) to a specific positive value. The value of the code is $-(5000 + n)$, where n is the value (always ≥ 0) assigned to the current line number. Action codes of this type are added to the list by the `\set@line@action` macro, as called to implement the `\advanceline` and `\setline` macros: this action only occurs when the user has specified some change to the line numbers using those macros. Normally `ledmac` computes the visible line numbers from the absolute line numbers with reference to the other action codes and the settings they invoke; it doesn't require an entry in the action-code list for every line.

Here are the commands to create these lists:

```

335     \list@create{\line@list}
336     \list@create{\insertlines@list}
337     \list@create{\actionlines@list}
338     \list@create{\actions@list}
339

```

```

\page@num We'll need some counts while we read the line-list, for the page number and the
\endpage@num ending page, line, and sub-line numbers. Some of these will be used again later
\endline@num on, when we are acting on the data in our list macros.
\endsubline@num
340 \newcount\page@num
341 \newcount\endpage@num
342 \newcount\endline@num
343 \newcount\endsubline@num

```

```

\ifnoteschanged@ If the number of footnotes in a section is different from what it was during the last
\noteschanged@true run, or if this is the very first time you've run LaTeX, on this file, the information
\noteschanged@false from the line-list used to place the notes will be wrong, and some notes will
                      probably be misplaced. When this happens, we prefer to give a single error message
                      for the whole section rather than messages at every point where we notice the
                      problem, because we don't really know where in the section notes were added or
                      removed, and the solution in any case is simply to run LaTeX two more times;

```

there's no fix needed to the document. The `\ifnoteschanged@` flag is set if such a change in the number of notes is discovered at any point.

```
344 \newif\ifnoteschanged@
```

19.4 Reading the line-list file

`\read@linelist` `\read@linelist{<file>}` is the control sequence that's called by `\beginnumbering` (via `\line@list@stuff`) to open and process a line-list file; its argument is the name of the file.

```
345 \newread\@inputcheck
346 \newcommand*\read@linelist}[1]{%
347   \list@clearing@reg
```

When the file is there we start a new group and make some special definitions we'll need to process it: it's a sequence of TeX commands, but they require a few special settings. We make `[` and `]` become grouping characters: they're used that way in the line-list file, because we need to write them out one at a time rather than in balanced pairs, and it's easier to just use something other than real braces. `@` must become a letter, since this is run in the ordinary LaTeX context. We ignore carriage returns, since if we're in horizontal mode they can get interpreted as spaces to be printed.

Our line, page, and line-locking counters were already zeroed by `\line@list@stuff` if this is being called from within `\beginnumbering`; sub-lineation will be turned off as well in that case. On the other hand, if this is being called from `\resumenumbering`, those things should still have the values they had when `\pausenumbering` was executed.

If the file is not there, we print an informative message.

Now, after these preliminaries, we start interpreting the file.

```
348   \get@linelistfile{#1}%
349   \endgroup
350
```

When the reading is done, we're all through with the line-list file. All the information we needed from it will now be encoded in our list macros.

Finally, we initialize the `\next@actionline` and `\next@action` macros, which specify where and what the next action to be taken is.

```
351   \global\page@num=\m@ne
352   \ifx\actionlines@list\empty
353     \gdef\next@actionline{1000000}%
354   \else
355     \glp\actionlines@list\to\next@actionline
356     \glp\actions@list\to\next@action
357   \fi}
358
```

`\list@clearing@reg` Clears the lists for `\read@linelist`

```
359 \newcommand*\list@clearing@reg}{%
```

```

360 \list@clear{\line@list}%
361 \list@clear{\insertlines@list}%
362 \list@clear{\actionlines@list}%
363 \list@clear{\actions@list}}

```

`\get@linelistfile` ledmac can take advantage of the LaTeX ‘safe file input’ macros to get the line-list file.

```

364 \newcommand*\get@linelistfile}[1]{%
365   \InputIfFileExists{#1}{%
366     \global\noteschanged@false
367     \begingroup
368       \catcode'\[=1 \catcode'\]=2
369       \makeatletter \catcode'\^M=9}{%
370     \led@warn@NoLineFile{#1}%
371     \global\noteschanged@true
372     \begingroup}%
373 }
374

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we’d have to do some file renaming outside of LaTeX for that to work. We’ve retained this slower approach to avoid that sort of hacking about, but have provided the `\pausenumbering` and `\resumenumbering` macros to help you if you run into macro memory limitations (see p. 10 above).

19.5 Commands within the line-list file

This section defines the commands that can appear within a line-list file. They all have very short names because we are likely to be writing very large numbers of them out. One macro, `\@l`, is especially short, since it will be written to the line-list file once for every line of text in a numbered section. (Another of these commands, `\@lab`, will be introduced in a later section, among the cross-referencing commands it is associated with.)

When these commands modify the various page and line counters, they deliberately do not say `\global`. This is because we want them to affect only the counter values within the current group when nested calls of `\@ref` occur. (The code assumes throughout that the value of `\globaldefs` is zero.)

The macros with **action** in their names contain all the code that modifies the action-code list: again, this is so that they can be turned off easily for nested calls of `\@ref`.

`\@l` `\@l` does everything related to the start of a new line of numbered text.
`\@l@reg`

In order to get the `\setlinenum` to work I had to slip in some new code at the start of the macro, to get the timing of the actions correct. The problem was that my original naive implementation of `\setlinenum` had a unfortunate tendency to change the number of the last line of the *preceding* paragraph. The new code is sort of based on the page number handling and `\setline`. It seems that a lot of fiddling with the line number internals is required.

In November 2004 in order to accurately determine page numbers I added these to the macro. It is now:

```
\@l{\page counter number}\{printed page number}
```

I don't (yet) use the printed number (i.e., the `\thepage`) but it may come in handy later. The macro `\fix@page` checks if a new page has started.

```
375 \newcommand{\@l}[2]{%
376   \fix@page{#1}%
377   \@l@reg}
378 \newcommand*{\@l@reg}{%
379   \ifx\l@dchset@num\relax \else
380     \advance\absline@num \@ne
381     \set@line@action
382     \let\l@dchset@num=\relax
383     \advance\absline@num \m@ne
384     \advance\line@num \m@ne
385   \fi
```

Now we are back to the original code.

First increment the absolute line-number, and perform deferred actions relating to page starts and sub-lines.

```
386 \advance\absline@num \@ne
387   \ifx\next@page@num\relax \else
388     \page@action
389     \let\next@page@num=\relax
390   \fi
391   \ifx\sub@change\relax \else
392     \ifnum\sub@change>\z@
393       \sublines@true
394     \else
395       \sublines@false
396     \fi
397     \sub@action
398     \let\sub@change=\relax
399   \fi
```

Fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```
400   \ifcase\@lock
401     \or
402       \@lock \tw@
403     \or \or
404       \@lock \z@
405   \fi
```

```

406      \ifcase\sub@lock
407      \or
408      \sub@lock \tw@
409      \or \or
410      \sub@lock \z@
411      \fi
      Now advance the visible line number, unless it's been locked.
412      \ifsublines@
413      \ifnum\sub@lock<\tw@
414      \advance\subline@num \@ne
415      \fi
416      \else
417      \ifnum\@lock<\tw@
418      \advance\line@num \@ne \subline@num \z@
419      \fi
420      \fi}
421

```

`\@page \@page{<num>}` marks the start of a new output page; its argument is the number of that page.

First we reset the visible line numbers, if we're numbering by page, and store the page number itself in a count.

```

422 \newcommand*{\@page}[1]{%
423   \ifbypage@
424   \line@num \z@ \subline@num \z@
425   \fi
426   \page@num=#1\relax

```

And we set a flag that tells `\@l` that a new page number is to be set, because other associated actions shouldn't occur until the next line-start occurs.

```

427   \def\next@page@num{#1}}
428

```

`\last@page@num \fix@page` basically replaces `\@page`. It determines whether or not a new page has been started, based on the page values held by `\@l`.

```

429 \newcount\last@page@num
430 \last@page@num=-10000
431 \newcommand*{\fix@page}[1]{%
432   \ifnum #1=\last@page@num
433   \else
434     \ifbypage@
435     \line@num=\z@ \subline@num=\z@
436     \fi
437     \page@num=#1\relax
438     \last@page@num=#1\relax
439     \def\next@page@num{#1}%
440   \fi}
441

```

`\@pend` These don't do anything at this point, but will have been added to the auxiliary file(s)
`\@pendR` if the ledpar package has been used. They are just here to stop ledmac from moaning
`\@lopL` if the ledpar is used for one run and then not for the following one.

```
\@lopR 442 \newcommand*{\@pend}[1]{%
443 \newcommand*{\@pendR}[1]{%
444 \newcommand*{\@lopL}[1]{%
445 \newcommand*{\@lopR}[1]{%
446
```

`\sub@on` The `\sub@on` and `\sub@off` macros turn sub-lineation on and off: but not directly,
`\sub@off` since such changes don't really take effect until the next line of text. Instead they
set a flag that notifies `\@l` of the necessary action.

```
447 \newcommand*{\sub@on}{\ifsublines@
448 \let\sub@change=\relax
449 \else
450 \def\sub@change{1}%
451 \fi}
452 \newcommand*{\sub@off}{\ifsublines@
453 \def\sub@change{-1}%
454 \else
455 \let\sub@change=\relax
456 \fi}
457
```

`\@adv` The `\@adv{<num>}` macro advances the current visible line number by the amount
specified as its argument. This is used to implement `\advanceLine`.

```
458 \newcommand*{\@adv}[1]{\ifsublines@
459 \advance\subline@num by #1\relax
460 \ifnum\subline@num<\z@
461 \led@warn@BadAdvancelineSubline
462 \subline@num \z@
463 \fi
464 \else
465 \advance\line@num by #1\relax
466 \ifnum\line@num<\z@
467 \led@warn@BadAdvancelineLine
468 \line@num \z@
469 \fi
470 \fi
471 \set@line@action}
472
```

`\@set` The `\@set{<num>}` macro sets the current visible line number to the value speci-
fied as its argument. This is used to implement `\setLine`.

```
473 \newcommand*{\@set}[1]{\ifsublines@
474 \subline@num=#1\relax
475 \else
476 \line@num=#1\relax
477 \fi
```

```
478 \set@line@action}
479
```

`\l@d@set` The `\l@d@set{<num>}` macro sets the line number for the next `\pstart...` to the value specified as its argument. This is used to implement `\setlinenum`.

`\l@dchset@num` is a flag to the `\@l` macro. If it is not `\relax` then a linenum change is to be done.

```
480 \newcommand*{\l@d@set}[1]{%
481   \line@num=#1\relax
482   \advance\line@num \@ne
483   \def\l@dchset@num{#1}}
484 \let\l@dchset@num\relax
485
```

`\page@action` `\page@action` adds an entry to the action-code list to change the page number.

```
486 \newcommand*{\page@action}{%
487   \xright@appenditem{\the\absline@num}\to\actionlines@list
488   \xright@appenditem{\next@page@num}\to\actions@list}
```

`\set@line@action` `\set@line@action` adds an entry to the action-code list to change the visible line number.

```
489 \newcommand*{\set@line@action}{%
490   \xright@appenditem{\the\absline@num}\to\actionlines@list
491   \ifsublines@
492     \@l@dttempcnta=-\subline@num
493   \else
494     \@l@dttempcnta=-\line@num
495   \fi
496   \advance\@l@dttempcnta by -5000
497   \xright@appenditem{\the\@l@dttempcnta}\to\actions@list}
```

`\sub@action` `\sub@action` adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the `\ifsublines@` flag.

```
498 \newcommand*{\sub@action}{%
499   \xright@appenditem{\the\absline@num}\to\actionlines@list
500   \ifsublines@
501     \xright@appenditem{-1001}\to\actions@list
502   \else
503     \xright@appenditem{-1002}\to\actions@list
504   \fi}
```

`\lock@on` `\lock@on` adds an entry to the action-code list to turn line number locking on.

`\do@lockon` The current setting of the sub-lineation flag tells us whether this applies to line numbers or sub-line numbers.

`\do@lockonL` Adding commands to the action list is slow, and it's very often the case that a lock-on command is immediately followed by a lock-off command in the line-list file, and therefore really does nothing. We use a look-ahead scheme here to detect such pairs, and add nothing to the line-list in those cases.

```

505 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
506 \newcommand*{\do@lockon}{%
507   \ifx\next\lock@off
508     \global\let\lock@off=\skip@lockoff
509   \else
510     \do@lockonL
511   \fi}
512 \newcommand*{\do@lockonL}{%
513   \xright@appenditem{\the\absline@num}\to\actionlines@list
514   \ifsublines@
515     \xright@appenditem{-1005}\to\actions@list
516     \ifnum\sub@lock=z@
517       \sub@lock \@ne
518     \else
519       \ifnum\sub@lock=\thr@@
520         \sub@lock \@ne
521       \fi
522     \fi
523   \else
524     \xright@appenditem{-1003}\to\actions@list
525     \ifnum\@lock=z@
526       \@lock \@ne
527     \else
528       \ifnum\@lock=\thr@@
529         \@lock \@ne
530       \fi
531     \fi
532   \fi}
533
\lock@off \lock@off adds an entry to the action-code list to turn line number locking off.
\do@lockoff 534 \newcommand*{\do@lockoffL}{%
\do@lockoffL 535 \xright@appenditem{\the\absline@num}\to\actionlines@list
\skip@lockoff 536 \ifsublines@
537   \xright@appenditem{-1006}\to\actions@list
538   \ifnum\sub@lock=\tw@
539     \sub@lock \thr@@
540   \else
541     \sub@lock \z@
542   \fi
543 \else
544   \xright@appenditem{-1004}\to\actions@list
545   \ifnum\@lock=\tw@
546     \@lock \thr@@
547   \else
548     \@lock \z@
549   \fi
550 \fi}
551 \newcommand*{\do@lockoff}{\do@lockoffL}
552 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}

```



```
553 \global\let\lock@off=\do@lockoff
554
```

`\n@num` This macro implements the `\skipnumbering` command. It uses a new action code, namely 1007.

```
555 \newcommand*{\n@num}{\n@num@reg}
556 \newcommand*{\n@num@reg}{%
557 \xright@appenditem{\the\absline@num}\to\actionlines@list
558 \xright@appenditem{-1007}\to\actions@list}
559
```

`\@ref` `\@ref` marks the start of a passage, for creation of a footnote reference. It takes two arguments:

- #1, the number of entries to add to `\insertlines@list` for this reference. This value, here and within `\edtext`, which computes it and writes it to the line-list file, will be stored in the count `\insert@count`.

```
560 \newcount\insert@count
```

- #2, a sequence of other line-list-file commands, executed to determine the ending line-number. (This may also include other `\@ref` commands, corresponding to uses of `\edtext` within the first argument of another instance of `\edtext`.)

`\dummy@ref` When nesting of `\@ref` commands does occur, it's necessary to temporarily redefine `\@ref` within `\@ref`, so that we're only doing one of these at a time.

```
561 \newcommand*{\dummy@ref}[2]{#2}
```

`\@ref@reg` The first thing `\@ref` (i.e. `\@ref@reg`) itself does is to add the specified number of items to the `\insertlines@list` list.

```
562 \newcommand*{\@ref}[2]{%
563 \@ref@reg{#1}{#2}}
564 \newcommand*{\@ref@reg}[2]{%
565 \global\insert@count=#1\relax
566 \loop\ifnum\insert@count>\z@
567 \xright@appenditem{\the\absline@num}\to\insertlines@list
568 \global\advance\insert@count \m@ne
569 \repeat
```

Next, process the second argument to determine the page and line numbers for the end of this lemma. We temporarily equate `\@ref` to a different macro that just executes its argument, so that nested `\@ref` commands are just skipped this time. Some other macros need to be temporarily redefined to suppress their action.

```
570 \begingroup
571 \let\@ref=\dummy@ref
572 \let\page@action=\relax
573 \let\sub@action=\relax
574 \let\set@line@action=\relax
```

```

575 \let\@lab=\relax
576 #2
577 \global\endpage@num=\page@num
578 \global\endline@num=\line@num
579 \global\endsubline@num=\subline@num
580 \endgroup

```

Now store all the information about the location of the lemma's start and end in `\line@list`.

```

581 \xright@appenditem%
582 {\the\page@num|\the\line@num|%
583 \ifsublines@ \the\subline@num \else 0\fi|%
584 \the\endpage@num|\the\endline@num|%
585 \ifsublines@ \the\endsubline@num \else 0\fi}}\to\line@list

```

Finally, execute the second argument of `\@ref` again, to perform for real all the commands within it.

```

586 #2}
587

```

19.6 Writing to the line-list file

We've now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we'll cover the commands that `ledmac` uses within the text of a section to write commands out to the line-list.

`\linenum@out` The file will be opened on output stream `\linenum@out`.

```
588 \newwrite\linenum@out
```

`\iffirst@linenum@out@` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open. The reason is that we want the output routine to write the page number for every page to this file; otherwise we'd have to write it at the start of every line. But it's not very easy for the output routine to tell whether an output stream is open or not. There's no way to test the status of a particular output stream directly, and the asynchronous nature of output routines makes the status hard to determine by other means.

We can manage pretty well by means of the `\iffirst@linenum@out@` flag; its inelegant name suggests the nature of the problem that made its creation necessary. It's set to be `true` before any `\linenum@out` file is opened. When such a file is opened for the first time, it's done using `\immediate`, so that it will at once be safe for the output routine to write to it; we then set this flag to `false`.

```

589 \newif\iffirst@linenum@out@
590 \iffirst@linenum@out@true

```

`\line@list@stuff` The `\line@list@stuff{⟨file⟩}` macro, which is called by `\beginnumbering`, performs all the line-list operations needed at the start of a section. Its argument is the name of the line-list file.

```
591 \newcommand*{\line@list@stuff}[1]{%
```

First, use the commands of the previous section to interpret the line-list file from the last run.

```
592 \read@linelist{#1}%
```

Now close the current output line-list file, if any, and open a new one. The first time we open a line-list file for output, we do it using `\immediate`, and clear the `\iffirst@linenum@out@` flag.

```
593 \iffirst@linenum@out@
594   \immediate\closeout\linenum@out
595   \global\first@linenum@out@false
596   \immediate\openout\linenum@out=#1\relax
597 \else
```

If we get here, then this is not the first line-list we've seen, so we don't open or close the files immediately.

```
598   \closeout\linenum@out
599   \openout\linenum@out=#1\relax
600 \fi}
601
```

`\new@line` The `\new@line` macro sends the `\@l` command to the line-list file, to mark the start of a new text line, and its page number.

```
602 \newcommand*{\new@line}{\write\linenum@out{\string\@l[\the\c@page][\thepage]}}
```

`\flag@start` We enclose a lemma marked by `\edtext` in `\flag@start` and `\flag@end`: these
`\flag@end` send the `\@ref` command to the line-list file. `\edtext` is responsible for setting the value of `\insert@count` appropriately; it actually gets done by the various footnote macros.

```
603 \newcommand*{\flag@start}{%
604   \edef\next{\write\linenum@out{%
605     \string\@ref[\the\insert@count] []}%
606   \next}
607 \newcommand*{\flag@end}{\write\linenum@out{[]}}
```

`\page@start` Originally the commentary was: `\page@start` writes a command to the line-list file noting the current page number; when used within an output routine, this should be called so as to place its `\write` within the box that gets shipped out, and as close to the top of that box as possible.

However, in October 2004 Alexej Krukov discovered that when processing long paragraphs that included Russian, Greek and Latin texts `ledmac` would go into an infinite loop, emitting thousands of blank pages. This was caused by being unable to find an appropriate place in the output routine. A different algorithm is now used for getting page numbers.

```
608 \newcommand*{\page@start}{%
609
```

`\startsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate instructions to the line-list file. When sub-lineation is in effect, the line number

counter is frozen and the sub-line counter advances instead. If one of these commands appears in the middle of a line, it doesn't take effect until the next line; in other words, a line is counted as a line or sub-line depending on what it started out as, even if that changes in the middle.

We tinker with `\lastskip` because a command of either sort really needs to be attached to the last word preceding the change, not the first word that follows the change. This is because sub-lineation will often turn on and off in mid-line—stage directions, for example, often are mixed with dialogue in that way—and when a line is mixed we want to label it using the system that was in effect at its start. But when sub-lineation begins at the very start of a line we have a problem, if we don't put in this code.

```
610 \newcommand*{\startsub}{\dimen0\lastskip
611   \ifdim\dimen0>0pt \unskip \fi
612   \write\linenum@out{\string\sub@on}%
613   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
614 \def\endsub{\dimen0\lastskip
615   \ifdim\dimen0>0pt \unskip \fi
616   \write\linenum@out{\string\sub@off}%
617   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
618
```

`\advanceline` You can use `\advanceline{<num>}` in running text to advance the current visible line-number by a specified value, positive or negative.

```
619 \newcommand*{\advanceline}[1]{\write\linenum@out{\string\@adv[#1]}}
```

`\setline` You can use `\setline{<num>}` in running text (i.e., within `\pstart... \pend`) to set the current visible line-number to a specified positive value.

```
620 \newcommand*{\setline}[1]{%
621   \ifnum#1<\z@
622     \led@warn@BadSetline
623   \else
624     \write\linenum@out{\string\@set[#1]}%
625   \fi}
626
```

`\setlinenum` You can use `\setlinenum{<num>}` before a `\pstart` to set the visible line-number to a specified positive value. It writes a `\l@d@set` command to the line-list file.

```
627 \newcommand*{\setlinenum}[1]{%
628   \ifnum#1<\z@
629     \led@warn@BadSetlinenum
630   \else
631     \write\linenum@out{\string\l@d@set[#1]}%
632   \fi}
633
```

`\startlock` You can use `\startlock` or `\endlock` in running text to start or end line number locking at the current line. They decide whether line numbers or sub-line numbers are affected, depending on the current state of the sub-lineation flags.

```

634 \newcommand*{\startlock}{\write\linenum@out{\string\lock@on}}
635 \def\endlock{\write\linenum@out{\string\lock@off}}
636

```

```

\ifl@dskipnumber In numbered text \skipnumbering will suspend the numbering for that particular
\l@dskipnumbertrue line.
\l@dskipnumberfalse
\skipnumbering 637 \newif\ifl@dskipnumber
\skipnumbering 638 \l@dskipnumberfalse
\skipnumbering@reg 639 \newcommand*{\skipnumbering}{\skipnumbering@reg}
640 \newcommand*{\skipnumbering@reg}{%
641 \write\linenum@out{\string\n@num}%
642 \advance\line{-1}}
643

```

20 Marking text for notes

The `\edtext` (or `\critext`) macro is used to create all footnotes and endnotes, as well as to print the portion of the main text to which a given note or notes is keyed. The idea is to have that lemma appear only once in the `.tex` file: all instances of it in the main text and in the notes are copied from that one appearance.

For convenience, I will use `*text` when I do not need to distinguish between `\edtext` and `\critext`. The `*text` macros take two arguments, the only difference between `\edtext` and `\critext` is how the second argument is delineated.

`\critext` requires two arguments. At any point within numbered text, you use it by saying:

```
\critext{#1}#2/
```

Similarly `\edtext` requires the same two arguments but you use it by saying:

```
\edtext{#1}{#2}
```

- `#1` is the piece of the main text being glossed; it gets added to the main text, and is also used as a lemma for notes to it.
- `#2` is a series of subsidiary macros that generate various kinds of notes. With `\critext` the `/` after `#2` *must* appear: it marks the end of the macro. (*The TeXbook*, p. 204, points out that when additional text to be matched follows the arguments like this, spaces following the macro are not skipped, which is very desirable since this macro will never be used except within text. Having an explicit terminator also helps keep things straight when nested calls to `\critext` are used.) Braces around `#2` are optional with `\critext` and required for `\edtext`.

The `*text` macro may be used (somewhat) recursively; that is, `*text` may be used within its own first argument. The code would be much simpler without this

feature, but nested notes will commonly be necessary: it's quite likely that we'll have an explanatory note for a long passage and notes on variants for individual words within that passage. The situation we can't handle is overlapping notes that aren't nested: for example, one note covering lines 10–15, and another covering 12–18. You can handle such cases by using the `\lemma` and `\linenum` macros within `#2`: they alter the copy of the lemma and the line numbers that are passed to the notes, and hence allow you to overcome any limitations of this system, albeit with extra effort.

The recursive operation of `*text` will fail if you try to use a copy that is called something other than `*text`. In order to handle recursion, `*text` needs to redefine its own definition temporarily at one point, and that doesn't work if the macro you are calling is not actually named `*text`. There's no problem as long as `*text` is not invoked in the first argument. If you want to call `*text` something else, it is best to create instead a macro that expands to an invocation of `*text`, rather than copying `*text` and giving it a new name; otherwise you will need to add an appropriate definition for your new macro to `\morenoexpands`.

Side effects of our line-numbering code make it impossible to use the usual footnote macros directly within a paragraph whose lines are numbered (see comments to `\do@line`, p.??). Instead, the appropriate note-generating command is appended to the list macro `\inserts@list`, and when `\pend` completes the paragraph it inserts all the notes at the proper places.

Note that we don't provide previous-note information, although it's often wanted; your own macros must handle that. We can't do it correctly without keeping track of what kind of notes have gone past: it's not just a matter of remembering the line numbers associated with the previous invocation of `*text`, because that might have been for a different kind of note. It is preferable for your footnote macros to store and recall this kind of information if they need it.

An example where some 'memory' of line numbers might be required is where there are several variant readings per line of text, and you do not wish the line number to be repeated for each lemma in the notes. After the first occurrence of the line number, you might want the symbol '||' instead of further occurrences, for instance. This can easily be done by a macro like `\printlines`, if it saves the last value of `\l@d@nums` that *it* saw, and then performs a simple conditional test to see whether to print a number or a '||'.

20.1 `\edtext` and `\critext` themselves

The various note-generating macros might want to request that commands be executed not at once, but in close connection with the start or end of the lemma. For example, footnote numbers in the text should be connected to the end of the lemma; or, instead of a single macro to create a note listing variants, you might want to use several macros in series to create individual variants, which would each add information to a private macro or token register, which in turn would be formatted and output when all of `#2` for the lemma has been read.

`\end@lemmas` To accomodate this, we provide a list macro to which macros may add commands

that should subsequently be executed at the end of the lemma when that lemma is added to the text of the paragraph. A macro should add its contribution to `\end@lemmas` by using `\xleft@appenditem`. (Anything that needs to be done at the *start* of the lemma may be handled using `\aftergroup`, since the commands specified within `\critext`'s second argument are executed within a group that ends just before the lemma is added to the main text.)

`\end@lemmas` is intended for the few things that need to be associated with the end of the lemma, like footnote numbers. Such numbers are not implemented in the current version, and indeed no use is currently made of `\end@lemmas` or of the `\aftergroup` trick. The general approach would be to define a macro to be used within the second argument of `\critext` that would add the appropriate command to `\end@lemmas`.

Commands that are added to this list should always take care not to do anything that adds possible line-breaks to the output; otherwise line numbering could be thrown off.

```
644 \list@create{\end@lemmas}
```

`\dummy@text` We now need to define a number of macros that allow us to weed out nested instances of `\critext`, and other problematic macros, from our lemma. This is similar to what we did in reading the line-list file using `\dummy@ref` and various redefinitions—and that's because nested `\critext` macros create nested `\@ref` entries in the line-list file.

Here's a macro that takes the same arguments as `\critext` but merely returns the first argument and ignores the second.

```
645 \long\def\dummy@text#1#2/{#1}
```

`\dummy@edtext` LaTeX users are not used to delimited arguments, so I provide a `\edtext` macro as well.

```
646 \newcommand{\dummy@edtext}[2]{#1}
```

We're going to need another macro that takes one argument and ignores it entirely. This is supplied by the LaTeX `\@gobble{<arg>}`.

`\no@expands` We need to turn off macro expansion for certain sorts of macros we're likely to see
`\morenoexpands` within the lemma and within the notes.

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.²³ This is done because we want to pass into our notes the generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

A second sort to turn off includes a few of the accent macros. Most are not a problem: an accent that's expanded to an `\accent` command may be harder to read but it works just the same. The ones that cause problems are: those that use

²³Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.

alignments— \TeX seems to get confused about the difference between alignment parameters and macro parameters; those that use temporary control sequences; and those that look carefully at what the current font is.

(The `\copyright` macro defined in `PLAIN \TeX` has this sort of problem as well, but isn't used enough to bother with. That macro, and any other that causes trouble, will get by all right if you put a `\protect` in front of it in your file.)

We also need to eliminate all `ledmac` macros like `\edlabel` and `\setline` that write things to auxiliary files: that writing should be done only once. And we make `\critext` itself, if it appears within its own argument, do nothing but copy its first argument.

Finally, we execute `\morenoexpands`. The version of `\morenoexpands` defined here does nothing; but you may define a version of your own when you need to add more expansion suppressions as needed with your macros. That makes it possible to make such additions without needing to copy or modify the standard `ledmac` code. If you define your own `\morenoexpands`, you must be very careful about spaces: if the macro adds any spaces to the text when it runs, extra space will appear in the main text when `\critext` is used.

(A related problem, not addressed by these two macros, is that of characters whose category code is changed by any the macros used in the arguments to `\critext`. Since the category codes are set when the arguments are scanned, macros that depend on changing them will not work. We have most often encountered this with characters that are made 'active' within text in some, but not all, of the languages used within the document. One way around the problem, if it takes this form, is to ensure that those characters are *always* active; within languages that make no special use of them, their associated control sequences should simply return the proper character.)

```

647 \newcommand*{\noexpands}{\let\rm=0\let\it=0\let\s1=0\let\bf=0\let\tt=0%
648   \let\b=0\let\c=0\let\d=0\let\t=0%
649   \let\select@lemmafnt=0%
650   \def\protect{\noexpand\protect\noexpand}%
651   \let\startsub=\relax \let\endsub=\relax
652   \let\startlock=\relax \let\endlock=\relax
653   \let\edlabel=\@gobble
654 % \let\edpageref=\@gobble
655 % \let\lineref=\@gobble
656 % \let\sublineref=\@gobble
657   \let\setline=\@gobble \let\advanceline=\@gobble
658   \let\critext=\dummy@text
659   \let\edtext=\dummy@edtext
660   \l@dtabnoexpands
661   \morenoexpands}
662 \let\morenoexpands=\relax
663

```

`\critext` Now we begin `\critext` itself. The definition requires a / after the arguments: this eliminates the possibility of problems about knowing where #2 ends. This

also changes the handling of spaces following an invocation of the macro: normally such spaces are skipped, but in this case they're significant because `#2` is a 'delimited parameter'. Since `\critext` is always used in running text, it seems more appropriate to pay attention to spaces than to skip them.

When executed, `\critext` first ensures that we're in horizontal mode.

```
664 \long\def\critext#1#2/{\leavevmode
```

`\@tag` Our normal lemma is just argument `#1`; but that argument could have further invocations of `\critext` within it. We get a copy of the lemma without any `\critext` macros within it by temporarily redefining `\critext` to just copy its first argument and ignore the other, and then expand `#1` into `\@tag`, our lemma.

This is done within a group that starts here, in order to get the original `\critext` restored; within this group we've also turned off the expansion of those control sequences commonly found within text that can cause trouble for us.

```
665 \begingroup
666 \no@expands
667 \xdef\@tag{\protect#1}%
```

`\l@d@nums` Prepare more data for the benefit of note-generating macros: the line references and font specifier for this lemma go to `\l@d@nums`.

```
668 \set@line
```

`\insert@count` will be altered by the note-generating macros: it counts the number of deferred footnotes or other insertions generated by this instance of `\critext`.

```
669 \global\insert@count=0
```

Now process the note-generating macros in argument `#2` (i.e., `\Afootnote`, `\lemma`, etc.). `\ignorespaces` is here to skip over any spaces that might appear at the start of `#2`; otherwise they wind up in the main text. Footnote and other macros that are used within `#2` should all end with `\ignorespaces` as well, to skip any spaces between macros when several are used in series.

```
670 \ignorespaces #2\relax
```

Finally, we're ready to admit the first argument into the current paragraph.

It's important that we generate and output all the notes for this chunk of text *before* putting the text into the paragraph: notes that are referenced by line number should generally be tied to the start of the passage they gloss, not the end. That should all be done within the expansion of `#2` above, or in `\aftergroup` commands within that expansion.

```
671 \flag@start
672 \endgroup
673 \showlemma{#1}%
```

Finally, we add any insertions that are associated with the *end* of the lemma. Footnotes that are identified by symbols rather than by where the lemma begins in the main text need to be done here, and not above.

```
674 \ifx\end@lemmas\empty \else
```

```

675 \glp\end@lemmas\to\x@lemma
676 \x@lemma
677 \global\let\x@lemma=\relax
678 \fi
679 \flag@end}

```

Here's the promised undelimited LaTeX version of `\critext`.

```

\edtext
680 \newcommand{\edtext}[2]{\leavevmode
681 \begin{group}
682 \noexpand
683 \xdef\@tag{\protect#1}%
684 \set@line
685 \global\insert@count=0
686 \ignorespaces #2\relax
687 \flag@start
688 \endgroup
689 \showlemma{#1}%
690 \ifx\end@lemmas\empty \else
691 \glp\end@lemmas\to\x@lemma
692 \x@lemma
693 \global\let\x@lemma=\relax
694 \fi
695 \flag@end}
696

```

`\ifnumberline` The `\ifnumberline` option can be set to `FALSE` to disable line numbering.

```

697 \newif\ifnumberline
698 \numberlinetrue
699 % \end{macrocode}
700 % \end{macro}
701 % \begin{macro}{\set@line}
702 % The \cs{set@line} macro
703 % is called by \cs{critext} to put the line-reference field and
704 % font specifier for the current block of text into \cs{l@d@nums}.
705 %
706 % One instance of \cs{critext} may generate several notes, or it
707 % may generate none---it's legitimate for argument \verb"#2" to \cs{critext} to
708 % be empty. But \cs{flag@start} and \cs{flag@end} induce the generation of
709 % a single entry in \cs{line@list} during the next run, and it's vital
710 % to also remove one and only one \cs{line@list} entry here.
711 % \begin{macrocode}
712 \newcommand*{\set@line}{%

```

If no more lines are listed in `\line@list`, something's wrong—probably just some change in the input. We set all the numbers to zeros, following an old publishing convention for numerical references that haven't yet been resolved.

```

713 \ifx\line@list\empty
714 \global\noteschanged@true

```

```

715 \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
716 \else
717 \gl@p\line@list\to\@tempb
718 \xdef\l@d@nums{\@tempb|\edfont@info}%
719 \global\let\@tempb=\undefined
720 \fi}
721

```

`\edfont@info` The macro `\edfont@info` returns coded information about the current font.

```

722 \newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}
723

```

20.2 Substitute lemma

`\lemma` The `\lemma{<text>}` macro allows you to change the lemma that's passed on to the notes.

```

724 \newcommand*{\lemma}[1]{\xdef\tag{\protect#1}\ignorespaces}

```

20.3 Substitute line numbers

`\linenum` The `\linenum` macro can change any or all of the page and line numbers that are passed on to the notes.

As argument `\linenum` takes a set of seven parameters separated by vertical bars, in the format used internally for `\l@d@nums` (see p.55): the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma. However, you can omit any parameters you don't want to change, and you can omit a string of vertical bars at the end of the argument. Hence `\linenum{18|4|0|18|7|1|0}` is an invocation that changes all the parameters, but `\linenum{|3}` only changes the starting line number, and leaves the rest unaltered.

We use `\\` as an internal separator for the macro parameters.

```

725 \newcommand*{\linenum}[1]{%
726 \xdef\@tempa{#1|||||\\noexpand\\l@d@nums}%
727 \global\let\l@d@nums=\empty
728 \expandafter\line@set\@tempa|\\ignorespaces}

```

`\line@set` `\linenum` calls `\line@set` to do the actual work; it looks at the first number in the argument to `\linenum`, sets the corresponding value in `\l@d@nums`, and then calls itself to process the next number in the `\linenum` argument, if there are more numbers in `\l@d@nums` to process.

```

729 \def\line@set#1|#2\\#3|#4\\{%
730 \gdef\@tempb{#1}%
731 \ifx\@tempb\empty
732 \l@d@add{#3}%
733 \else
734 \l@d@add{#1}%
735 \fi

```

```

736 \gdef\@tempb{#4}%
737 \ifx\@tempb\empty\else
738     \l@d@add{|\line@set#2\#4\}%
739 \fi}

```

`\l@d@add` `\line@set` uses `\l@d@add` to tack numbers or vertical bars onto the right hand end of `\l@d@nums`.

```

740 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
741

```

21 Paragraph decomposition and reassembly

In order to be able to count the lines of text and affix line numbers, we add an extra stage of processing for each paragraph. We send the paragraph into a box register, rather than straight onto the vertical list, and when the paragraph ends we slice the paragraph into its component lines; to each line we add any notes or line numbers, add a command to write to the line-list, and then at last send the line to the vertical list. This section contains all the code for this processing.

21.1 Boxes, counters, `\pstart` and `\pend`

<pre> \raw@text \ifnumberedpar@ \numberedpar@true \numberedpar@false \num@lines \one@line \par@line </pre>	<p>Here are numbers and flags that are used internally in the course of the paragraph decomposition.</p> <p>When we first form the paragraph, it goes into a box register, <code>\raw@text</code>, instead of onto the current vertical list. The <code>\ifnumberedpar@</code> flag will be <code>true</code> while a paragraph is being processed in that way. <code>\num@lines</code> will store the number of lines in the paragraph when it's complete. When we chop it up into lines, each line in turn goes into the <code>\one@line</code> register, and <code>\par@line</code> will be the number of that line within the paragraph.</p>
--	--

```

742 \newbox\raw@text
743 \newif\ifnumberedpar@
744 \newcount\num@lines
745 \newbox\one@line
746 \newcount\par@line

```

<pre> \pstart numberpstarttrue numberpstartfalse thepstart \firstpstart </pre>	<p><code>\pstart</code> starts the paragraph by clearing the <code>\inserts@list</code> list and other relevant variables, and then arranges for the subsequent text to go into the <code>\raw@text</code> box. <code>\pstart</code> needs to appear at the start of every paragraph that's to be numbered; the <code>\autopar</code> command below may be used to insert these commands automatically.</p>
--	---

Beware: everything that occurs between `\pstart` and `\pend` is happening within a group; definitions must be global if you want them to survive past the end of the paragraph.

You can use the command `\numberpstarttrue` to insert a number on every `\pstart`. To stop the numbering, you must use `\numberpstartfalse`. To reset the numbering of `\pstarts`, insert

```

\setcounter{pstart}{0}

747
748 \newcounter{pstart}
749 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
750 \newif\ifnumberpstart
751 \newif\iffirstpstart
752 \firstpstarttrue
753 \numberpstartfalse
754 \newcommand*{\pstart}{
755   \ifbypstart@\iffirstpstart
756     \firstpstartfalse
757   \else
758     \setline{0}
759   \fi
760 \fi
761 \if@nobreak
762 \let\@oldnobreak\@nobreaktrue
763 \else
764 \let\@oldnobreak\@nobreakfalse
765 \fi
766 \@nobreaktrue
767 \ifnumbering \else
768   \led@err@PstartNotNumbered
769   \beginnumbering
770 \fi
771 \ifnumberedpar@
772   \led@err@PstartInPstart
773 \pend
774 \fi
775 \list@clear{\inserts@list}%
776 \global\let\next@insert=\empty
777 \begingroup\normal@pars
778 \global\setbox\raw@text=\vbox\bgroup\ifautopar\else\ifnumberpstart\ifinstanza\else\ifsidepstartnum\
779 \numberedpar@true}

```

`\pend` `\pend` must be used to end a numbered paragraph.

```

780 \newcommand*{\pend}{\ifnumbering \else
781   \led@err@PendNotNumbered
782 \fi
783 \ifnumberedpar@ \else
784   \led@err@PendNoPstart
785 \fi

```

We set all the usual interline penalties to zero and then immediately call `\endgraf` to end the paragraph; this ensures that there'll be no large interline penalties to prevent us from slicing the paragraph into pieces. These penalties revert to the values that you set when the group for the `\vbox` ends. Then we call `\do@line`

to slice a line off the top of the paragraph, add a line number and footnotes, and restore it to the page; we keep doing this until there aren't any more lines left.

```

786 \l@dzzeropenalties
787 \endgraf\global\num@lines=\prevgraf\egroup
788 \global\par@line=0
789 \loop\ifvbox\raw@text
790   \do@line
791 \repeat

```

Deal with any leftover notes, and then end the group that was begun in the `\pstart`.

```

792 \flush@notes
793 \endgroup
794 \ignorespaces
795 \ifnumberpstart
796 \addtocounter{pstart}{1}
797 \pstartnumtrue
798 \fi
799 \@oldnobreak}
800

```

`\l@dzzeropenalties` A macro to zero penalties for `\pend`.

```

801 \newcommand*{\l@dzzeropenalties}{%
802   \brokenpenalty \z@ \clubpenalty \z@
803   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
804   \postdisplaypenalty \z@ \widowpenalty \z@}
805

```

`\autopar` In most cases it's only an annoyance to have to label the paragraphs to be numbered with `\pstart` and `\pend`. `\autopar` will do that automatically, allowing you to start a paragraph with its first word and no other preliminaries, and to end it with a blank line or a `\par` command. The command should be issued within a group, after `\beginnumbering` has been used to start the numbering; all paragraphs within the group will be affected.

A few situations can cause problems. One is a paragraph that begins with a begin-group character or command: `\pstart` will not get invoked until after such a group beginning is processed; as a result the character that ends the group will be mistaken for the end of the `\vbox` that `\pstart` creates, and the rest of the paragraph will not be numbered. Such paragraphs need to be started explicitly using `\indent`, `\noindent`, or `\leavevmode`—or `\pstart`, since you can still include your own `\pstart` and `\pend` commands even with `\autopar` on.

Prematurely ending the group within which `\autopar` is in effect will cause a similar problem. You must either leave a blank line or use `\par` to end the last paragraph before you end the group.

The functioning of this macro is more tricky than the usual `\everypar`: we don't want anything to go onto the vertical list at all, so we have to end the paragraph, erase any evidence that it ever existed, and start it again using `\pstart`. We remove the paragraph-indentation box using `\lastbox` and save the width,

and then skip backwards over the `\parskip` that's been added for this paragraph. Then we start again with `\pstart`, restoring the indentation that we saved, and locally change `\par` so that it'll do our `\pend` for us.

```

806 \newif\ifautopar
807 \autoparfalse
808 \newcommand*{\autopar}{
809   \ifledRcol
810     \ifnumberingR \else
811       \led@err@AutoparNotNumbered
812     \beginnumberingR
813     \fi
814   \else
815     \ifnumbering \else
816       \led@err@AutoparNotNumbered
817     \beginnumbering
818     \fi
819   \fi
820   \autopartrue
821   \everypar={\setbox0=\lastbox
822     \endgraf \vskip-\parskip
823     \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\fi\fi
824     \let\par=\pend}%
825   \ignorespaces}

```

`\normal@pars` We also define a macro which we can rely on to turn off the `\autopar` definitions at various important places, if they are in force. We'll want to do this within footnotes, for example.

```

826 \newcommand*{\normal@pars}{\everypar={}\let\par\endgraf}
827

```

21.2 Processing one line

`\do@line` The `\do@line` macro is called by `\pend` to do all the processing for a single line of text.

```

828 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
829 \newcommand*{\do@line}{%
830   {\vbadness=10000
831     \splittopskip=\z@
832     \do@linehook
833   \l@emptyd@ta
834     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
835   \unvbox\one@line \global\setbox\one@line=\lastbox
836   \getline@num
837   \affixline@num
838   \affixpstart@num
839   \hb@xt@ \linewidth{\inserthangingsymbol\l@dld@ta\add@inserts\affixside@note
840     \l@dlsn@te
841     {\ledllfill\hb@xt@ \wd\one@line{\new@line\l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%

```

```

842     \l@drsn@te
843   }}%
844 %\end{macrocode}
845 % The \cs{prevline@num} counter is used to check the insertion of hanging symbol.
846 %\begin{macrocode}
847   \prevline@num=\line@num%
848   \add@penalties}
849
\do@linehook A hook into \do@line.
850 \newcommand*{\do@linehook}{}

\l@emptyd@ta Nulls the \l@...d@ta, which may later hold line numbers. Similarly for \l@dcsnotetext
\l@dld@ta for the text of a sidenote.
\l@drd@ta 851 \newcommand*{\l@emptyd@ta}{%
\l@dcsnotetext 852   \gdef\l@dld@ta{}%
853   \gdef\l@drd@ta{}%
854   \gdef\l@dcsnotetext{}}
855

\l@dlsn@te Zero width boxes of the left and right side notes, together with their kerns.
\l@drsn@te 856 \newcommand{\l@dlsn@te}{%
857   \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
858 \newcommand{\l@drsn@te}{%
859   \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
860

\ledllfill These macros are called at the left (\ledllfill) and the right (\ledrlfill) of each
\ledrlfill numbered line. The initial definitions correspond to the original code for \do@line.
861 \newcommand*{\ledllfill}{\hfil}
862 \newcommand*{\ledrlfill}{}
863

```

21.3 Line and page number computation

```

\getline@num The \getline@num macro determines the page and line numbers for the line we're
about to send to the vertical list.
864 \newcommand*{\getline@num}{%
865   \ifnumberline
866     \global\advance\absline@num \@ne
867   \fi
868   \do@actions
869   \do@ballast
870   \ifnumberline
871     \ifsublines@
872       \ifnum\sub@lock<\tw@
873         \global\advance\subline@num \@ne
874       \fi

```



```

875 \else
876   \ifnum\@lock<\tw@
877     \global\advance\line@num \@ne
878     \global\subline@num \z@
879   \fi
880 \fi
881 \fi
882 }

```

`\do@ballast` The real work in the macro above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballast` out of the way. This macro looks to see if there is an action to be performed on the *next* line, and if it is going to be a page break action, `\do@ballast` decreases the count `\ballast@count` counter by the amount of `ballast`. This means, in practice, that when `\add@penalties` assigns penalties at this point, `TEX` will be given extra encouragement to break the page here (see p. 89).

`\ballast@count` First we set up the required counters; they are initially set to zero, and will remain
`\c@ballast` so unless you say `\setcounter{ballast}{<some figure>}` in your document.

```

883 \newcount\ballast@count
884 \newcounter{ballast}
885 \setcounter{ballast}{0}

```

And here is `\do@ballast` itself. It advances `\absline@num` within the protection of a group to make its check for what happens on the next line.

```

886 \newcommand*\do@ballast{\global\ballast@count \z@
887 \begingroup
888   \advance\absline@num \@ne
889   \ifnum\next@actionline=\absline@num
890     \ifnum\next@action>-1001\relax
891     \global\advance\ballast@count by -\c@ballast
892   \fi
893 \fi
894 \endgroup}

```

`\do@actions` The `\do@actions` macro looks at the list of actions to take at particular absolute
`\do@actions@next` line numbers, and does everything that's specified for the current line.

It may call itself recursively, and to do this efficiently (using `TEX`'s optimization for tail recursion), we define a control-sequence called `\do@actions@next` that is always the last thing that `\do@actions` does. If there could be more actions to process for this line, `\do@actions@next` is set equal to `\do@actions`; otherwise it's just `\relax`.

```

895 \newcommand*\do@actions{%
896   \global\let\do@actions@next=\relax
897   \ifnum\absline@num<\next@actionline\else

```

First, page number changes, which will generally be the most common actions. If we're restarting lineation on each page, this is where it happens.

```

898   \ifnum\next@action>-1001

```

```

899     \global\page@num=\next@action
900     \ifbypage@
901         \global\line@num=\z@ \global\subline@num=\z@
902     \fi

```

Next, we handle commands that change the line-number values. (We subtract 5001 rather than 5000 here because the line number is going to be incremented automatically in `\getline@num`.)

```

903     \else
904         \ifnum\next@action<-4999
905             \@l@dttempcnta=-\next@action
906             \advance\@l@dttempcnta by -5001
907             \ifsublines@
908                 \global\subline@num=\@l@dttempcnta
909             \else
910                 \global\line@num=\@l@dttempcnta
911             \fi

```

It's one of the fixed codes. We rescale the value in `\@l@dttempcnta` so that we can use a case statement.

```

912         \else
913             \@l@dttempcnta=-\next@action
914             \advance\@l@dttempcnta by -1000
915             \do@actions@fixedcode
916         \fi
917     \fi

```

Now we get information about the next action off the list, and then set `\do@actions@next` so that we'll call ourself recursively: the next action might also be for this line.

There's no warning if we find `\actionlines@list` empty, since that will always happen near the end of the section.

```

918     \ifx\actionlines@list\empty
919         \gdef\next@actionline{1000000}%
920     \else
921         \gl@p\actionlines@list\to\next@actionline
922         \gl@p\actions@list\to\next@action
923         \global\let\do@actions@next=\do@actions
924     \fi
925 \fi

```

Make the recursive call, if necessary.

```

926 \do@actions@next}
927

```

`\do@actions@fixedcode` This macro handles the fixed codes for `\do@actions`. It is one big case statement.

```

928 \newcommand*{\do@actions@fixedcode}{%
929     \ifcase\@l@dttempcnta
930     \or%                % 1001

```

```

931 \global\sublines@true
932 \or% % 1002
933 \global\sublines@false
934 \or% % 1003
935 \global\@lock=\@ne
936 \or% % 1004
937 \ifnum\@lock=\tw@
938 \global\@lock=\thr@@
939 \else
940 \global\@lock=\z@
941 \fi
942 \or% % 1005
943 \global\sub@lock=\@ne
944 \or% % 1006
945 \ifnum\sub@lock=\tw@
946 \global\sub@lock=\thr@@
947 \else
948 \global\sub@lock=\z@
949 \fi
950 \or% % 1007
951 \l@dskipnumbertrue
952 \else
953 \led@warn@BadAction
954 \fi}
955
956

```

21.4 Line number printing

`\affixline@num` `\affixline@num` originally took a single argument, a series of commands for printing the line just split off by `\do@line`; it put that line back on the vertical list, and added a line number if necessary. It now just puts a left line number into `\l@dld@ta` or a right line number into `\l@drd@ta` if required.

To determine whether we need to affix a line number to this line, we compute the following:

$$\begin{aligned}
 n &= \text{int}((\text{linenum} - \text{firstlinenum}) / \text{linenumincrement}) \\
 m &= \text{firstlinenum} + (n \times \text{linenumincrement})
 \end{aligned}$$

(where *int* truncates a real number to an integer). *m* will be equal to *linenum* only if we're to paste a number on here. However, the formula breaks down for the first line to number (and any before that), so we check that case separately: if `\line@num ≤ \firstlinenum`, we compare the two directly instead of making these calculations.

We compute, in the scratch counter `\@l@dtmpcnta`, the number of the next line that should be printed with a number (*m* in the above discussion), and move the current line number into the counter `\@l@dtmpcntb` for comparison.

Remember that some counts are now counters!

First, the case when we're within a sub-line range.

```
957 \newcommand*{\affixline@num}{%
```

No number is attached if `\ifl@dskipnumber` is TRUE (and then it is set to its normal FALSE value). No number is attached if `\ifnumberline` is FALSE (the normal value is TRUE).

```
958 \ifnumberline
959 \ifl@dskipnumber
960   \global\l@dskipnumberfalse
961 \else
962   \ifsublines@
963     \l@dttempcntb=\subline@num
964     \ifnum\subline@num>\c@firstsublinenum
965       \l@dttempcnta=\subline@num
966       \advance\l@dttempcnta by-\c@firstsublinenum
967       \divide\l@dttempcnta by\c@sublinenumincrement
968       \multiply\l@dttempcnta by\c@sublinenumincrement
969       \advance\l@dttempcnta by\c@firstsublinenum
970     \else
971       \l@dttempcnta=\c@firstsublinenum
972     \fi
```

That takes care of computing the values for comparison, but if line number locking is in effect we have to make a further check. If this check fails, then we disable the line-number display by setting the counters to arbitrary but unequal values.

```
973   \ch@cksub@l@ck
```

Now the line number case, which works the same way.

```
974   \else
975     \l@dttempcntb=\line@num
```

Check on the `\linenumberlist` If it's `\empty` use the standard algorithm.

```
976   \ifx\linenumberlist\empty
977     \ifnum\line@num>\c@firstlinenum
978       \l@dttempcnta=\line@num
979       \advance\l@dttempcnta by-\c@firstlinenum
980       \divide\l@dttempcnta by\c@linenumincrement
981       \multiply\l@dttempcnta by\c@linenumincrement
982       \advance\l@dttempcnta by\c@firstlinenum
983     \else
984       \l@dttempcnta=\c@firstlinenum
985     \fi
986   \else
```

The `\linenumberlist` wasn't `\empty`, so here's Wayne's numbering mechanism. This takes place in TeX's mouth.

```
987     \l@dttempcnta=\line@num
988     \edef\rem@inder{\linenumberlist,\number\line@num,}%
989     \edef\sc@n@list{\def\noexpand\sc@n@list
990       ###1,\number\l@dttempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
```

```

991      \sc@n@list\expandafter\sc@n@list\rem@inder|%
992      \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
993  \fi

```

A locking check for lines, just like the version for sub-line numbers above.

```

994  \ch@ck@l@ck
995  \fi

```

The following test is true if we need to print a line number.

```

996  \ifnum\@l@tempcnta=\@l@tempcntb

```

If we got here, we're going to print a line number; so now we need to calculate a number that will tell us which side of the page will get the line number. We start from `\line@margin`, which asks for one side always if it's less than 2; and then if the side does depend on the page number, we simply add the page number to this side code—because the values of `\line@margin` have been devised so that this produces a number that's even for left-margin numbers and odd for right-margin numbers.

For LaTeX we have to consider two column documents as well. In this case I think we need to put the numbers at the outside of the column — the left of the first column and the right of the second. Do the twocolumn stuff before going on with the original code.

`\l@dld@ta` A left line number is stored in `\l@dld@ta` and a right one in `\l@drd@ta`.

```

\l@drd@ta 997  \if@twocolumn
998      \if@firstcolumn
999      \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1000     \else
1001     \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1002     \fi
1003 \else

```

Continuing the original code ...

```

1004  \@l@tempcntb=\line@margin
1005  \ifnum\@l@tempcntb>\@ne
1006      \advance\@l@tempcntb \page@num
1007  \fi

```

Now print the line (#1) with its page number.

```

1008  \ifodd\@l@tempcntb
1009  \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1010  \else
1011  \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1012  \fi
1013 \fi
1014 \else

```

As no line number is to be appended, we just print the line as is.

```

1015 %%      #1%
1016 \fi

```

Now fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```
1017 \fix@locks
1018 \fi
1019 \fi
1020 }
1021
```

`\ch@cksub@l@ck` These macros handle line number locking for `\affixline@num`. `\ch@cksub@l@ck` checks subline locking. If it fails, then we disable the line-number display by setting the counters to arbitrary but unequal values.

```
1022 \newcommand*{\ch@cksub@l@ck}{%
1023   \ifcase\sub@lock
1024     \or
1025       \ifnum\sublock@disp=\@ne
1026         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1027       \fi
1028     \or
1029       \ifnum\sublock@disp=\tw@ \else
1030         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1031       \fi
1032     \or
1033       \ifnum\sublock@disp=\z@
1034         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1035       \fi
1036   \fi}
```

Similarly for line numbers.

```
1037 \newcommand*{\ch@ck@l@ck}{%
1038   \ifcase\@lock
1039     \or
1040       \ifnum\lock@disp=\@ne
1041         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1042       \fi
1043     \or
1044       \ifnum\lock@disp=\tw@ \else
1045         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1046       \fi
1047     \or
1048       \ifnum\lock@disp=\z@
1049         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
1050       \fi
1051   \fi}
```

Fix the lock counters. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```
1052 \newcommand*{\fix@locks}{%
1053   \ifcase\@lock
1054     \or
1055     \global\@lock=\tw@
```

```

1056 \or \or
1057   \global\@lock=\z@
1058 \fi
1059 \ifcase\sub@lock
1060 \or
1061   \global\sub@lock=\tw@
1062 \or \or
1063   \global\sub@lock=\z@
1064 \fi}
1065

```

`\pageparbreak` Because of TeX's asynchronous page breaking mechanism we can never be sure juust where it will make a break and, naturally, it has already decided exactly how it will typeset any remainder of a paragraph that crosses the break. This is disconcerting when trying to number lines by the page or put line numbers in different margins. This macro tries to force an invisible paragraph break and a page break.

```

1066 \newcommand{\pageparbreak}{\pend\newpage\pstart\noindent}
1067

```

21.5 Pstart number printing in side

In side, the printing of pstart number is running like the printing of line number. There is only some differences :

- The pstarts counter is upgrade in the `\pend` command. Consequently, the `\affixpstart@num` command has not to upgrade it, unlike the `\affixline@num` which upgrades the lines counter.
- To print the pstart number only at the begining of a pstart, and not in every line, a boolean test is made. The `\pstartnum` boolean is set to TRUE at every `\pend`. It's tried in the `\leftpstartnum` and `\rightpstartnum` commands. After the try, it is set to FALSE.

```

\leftpstartnum
\rightpstartnum 1068
\ifsidepstartnum 1069 \newif\ifsidepstartnum
1070 \newcommand*{\affixpstart@num}{%
1071   \ifsidepstartnum
1072     \if@twocolumn
1073       \if@firstcolumn
1074         \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
1075       \else
1076         \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
1077       \fi
1078     \else
1079       \@l@tempcntb=\line@margin
1080       \ifnum\@l@tempcntb>\@ne
1081         \advance\@l@tempcntb \page@num
1082       \fi

```

```

1083         \ifodd\@l@dttempcntb
1084             \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
1085         \else
1086             \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
1087         \fi
1088     \fi
1089 \fi
1090
1091 }
1092 %
1093
1094 \newif\ifpstartnum
1095 \pstartnumtrue
1096 \newcommand*{\leftpstartnum}{
1097     \ifpstartnum\thepstart
1098     \kern\linenumsep\fi
1099     \global\pstartnumfalse
1100 }
1101 \newcommand*{\rightpstartnum}{
1102     \ifpstartnum
1103     \kern\linenumsep
1104     \thepstart
1105     \fi
1106     \global\pstartnumfalse
1107 }

```

21.6 Add insertions to the vertical list

`\inserts@list` `\inserts@list` is the list macro that contains the inserts that we save up for one paragraph.

```
1108 \list@create{\inserts@list}
```

`\add@inserts` `\add@inserts` is the penultimate macro used by `\do@line`; it takes insertions saved in a list macro and sends them onto the vertical list.

It may call itself recursively, and to do this efficiently (using T_EX's optimization for tail recursion), we define a control-sequence called `\add@inserts@next` that is always the last thing that `\add@inserts` does. If there could be more inserts to process for this line, `\add@inserts@next` is set equal to `\add@inserts`; otherwise it's just `\relax`.

```

1109 \newcommand*{\add@inserts}{%
1110     \global\let\add@inserts@next=\relax

```

If `\inserts@list` is empty, there aren't any more notes or insertions for this paragraph, and we needn't waste our time.

```
1111 \ifx\inserts@list\empty \else
```

The `\next@insert` macro records the number of the line that receives the next footnote or other insert; it's empty when we start out, and just after we've affixed a note or insert.


```

1112 \ifx\next@insert\empty
1113 \ifx\insertlines@list\empty
1114 \global\noteschanged@true
1115 \gdef\next@insert{100000}%
1116 \else
1117 \gl@p\insertlines@list\to\next@insert
1118 \fi
1119 \fi

```

If the next insert's for this line, tack it on (and then erase the contents of the insert macro, as it could be quite large). In that case, we also set `\add@inserts@next` so that we'll call ourself recursively: there might be another insert for this same line.

```

1120 \ifnum\next@insert=\absline@num
1121 \gl@p\inserts@list\to\@insert
1122 \@insert
1123 \global\let\@insert=\undefined
1124 \global\let\next@insert=\empty
1125 \global\let\add@inserts@next=\add@inserts
1126 \fi
1127 \fi

```

Make the recursive call, if necessary.

```

1128 \add@inserts@next}
1129

```

21.7 Penalties

`\add@penalties` `\add@penalties` is the last macro used by `\do@line`. It adds up the club, widow, and interline penalties, and puts a single penalty of the appropriate size back into the paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

In this code, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we're working on at the moment. The count `\@l@tempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast` above (p.81). Finally, the penalty is checked to see that it doesn't go below -10000 .

```

1130 \newcommand*{\add@penalties}{\@l@tempcnta=\ballast@count
1131 \ifnum\num@lines>\@ne
1132 \global\advance\par@line \@ne
1133 \ifnum\par@line=\@ne
1134 \advance\@l@tempcnta \clubpenalty
1135 \fi
1136 \@l@tempcntb=\par@line \advance\@l@tempcntb \@ne
1137 \ifnum\@l@tempcntb=\num@lines
1138 \advance\@l@tempcnta \widowpenalty
1139 \fi

```

```

1140 \ifnum\par@line<\num@lines
1141 \advance\@l@tempcnta \interlinepenalty
1142 \fi
1143 \fi
1144 \ifnum\@l@tempcnta=\z@
1145 \relax
1146 \else
1147 \ifnum\@l@tempcnta>-10000
1148 \penalty\@l@tempcnta
1149 \else
1150 \penalty -10000
1151 \fi
1152 \fi}
1153

```

21.8 Printing leftover notes

`\flush@notes` The `\flush@notes` macro is called after the entire paragraph has been sliced up and sent on to the vertical list. If the number of notes to this paragraph has increased since the last run of \TeX , then there can be leftover notes that haven't yet been printed. An appropriate error message will be printed elsewhere; but it's best to go ahead and print these notes somewhere, even if it's not in quite the right place. What we do is dump them all out here, so that they should be printed on the same page as the last line of the paragraph. We can hope that's not too far from the proper location, to which they'll move on the next run.

```

1154 \newcommand*{\flush@notes}{%
1155 \xloop
1156 \ifx\inserts@list\empty \else
1157 \gl@p\inserts@list\to\@insert
1158 \@insert
1159 \global\let\@insert=\undefined
1160 \repeat}
1161

```

`\xloop` `\xloop` is a variant of the PLAIN \TeX `\loop` macro, useful when it's hard to construct a positive test using the \TeX `\if` commands—as in `\flush@notes` above. One says `\xloop ... \if ... \else ... \repeat`, and the action following `\else` is repeated as long as the `\if` test fails. (This macro will work wherever the PLAIN \TeX `\loop` is used, too, so we could just call it `\loop`; but it seems preferable not to change the definitions of any of the standard macros.)

This variant of `\loop` was introduced by Alois Kabelschacht in *TUGboat* 8 (1987), pp. 184–5.

```

1162 \def\xloop#1\repeat{%
1163 \def\body{#1\expandafter\body\fi}%
1164 \body}
1165

```

22 Footnotes

The footnote macros are adapted from those in `PLAIN TEX`, but they differ in these respects: the outer-level commands must add other commands to a list macro rather than doing insertions immediately; there are five separate levels of footnotes, not just one; and there are options to reformat footnotes into paragraphs or into multiple columns.

22.1 Fonts

Before getting into the details of formatting the notes, we set up some font macros. It is the notes that present the greatest challenge for our font-handling mechanism, because we need to be able to take fragments of our main text and print them in different forms: it is common to reduce the size, for example, without otherwise changing the fonts used.

I have deleted all Plain Font-related code and just kept the code for NFSS font handling.

`\notefontsetup` The font setup defined in `\notefontsetup` defines the standard fonts for the text of the footnotes. Parts of the footnote, such as the line number references and the lemma, are enclosed in groups, with their own font macros, so a note in plain roman can still have line numbers in bold, say, and the lemma in the same font encoding, family, series, and shape of font as in the main text. Typically this definition should specify only a size.

The original font for `\notefontsetup` effectively maps to LaTeX `\footnotesize` for a 10pt document.

```
1166 \newcommand*{\notefontsetup}{\footnotesize}
```

`\notenumfont` The line numbers will be printed using the font selected by executing `\notenumfont`.

The original font for `\notenumfont` maps to LaTeX `\scriptsize` for a 10pt document. However, the description in the user guide does not seem to match the definition (the usage guide says that the size is `\notefontsetup`).

```
1167 \newcommand*{\notenumfont}{\normalfont}
```

`\select@lemmafонт` `\select@lemmafонт` is provided to set the right font for the lemma in a note.

`\select@@lemmafонт` This macro extracts the font specifier from the line and page number cluster, and issues the associated font-changing command, so that the lemma is printed in its original font.

```
1168 \def\select@lemmafонт#1|#2|#3|#4|#5|#6|#7|{\select@@lemmafонт#7|}%
1169 \def\select@@lemmafонт#1/#2/#3/#4|{%
1170   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
1171    \selectfont}%
1172
```

22.2 Outer-level footnote commands

\Afootnote The outer-level footnote commands will look familiar: they're just called **\Afootnote**, **\Bfootnote**, etc., instead of plain **\footnote**. What they do, however, is quite different, since they have to operate in conjunction with **\critext** when numbering is in effect.

If we're within a line-numbered paragraph, then, we tack this note onto the **\inserts@list** list, and increment the deferred-page-bottom-note counter.

```
1173 \newcommand*{\Afootnote}[1]{%
1174   \ifnumberedpar@
1175     \xright@appenditem{\noexpand\Afootnote{A}%
1176                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1177     \global\advance\insert@count \@ne
```

Within free text, there's no need to put off making the insertion for this note. No line numbers are available, so this isn't generally that useful; but you might want to use it to get around some limitation of **ledmac**.

```
1178   \else
1179     \Afootnote{A}{\l@d@nums}{\@tag}{#1}%
1180   \fi\ignorespaces}
```

\Bfootnote We need similar commands for the other footnote series.

```
\Cfootnote 1181 \newcommand*{\Bfootnote}[1]{%
\Bfootnote 1182   \ifnumberedpar@
\Bfootnote 1183     \xright@appenditem{\noexpand\Bfootnote{B}%
1184                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1185     \global\advance\insert@count \@ne
1186   \else
1187     \Bfootnote{B}{\l@d@nums}{\@tag}{#1}%
1188   \fi\ignorespaces}

1189 \newcommand*{\Cfootnote}[1]{%
1190   \ifnumberedpar@
1191     \xright@appenditem{\noexpand\Cfootnote{C}%
1192                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1193     \global\advance\insert@count \@ne
1194   \else
1195     \Cfootnote{C}{\l@d@nums}{\@tag}{#1}%
1196   \fi\ignorespaces}

1197 \newcommand*{\Dfootnote}[1]{%
1198   \ifnumberedpar@
1199     \xright@appenditem{\noexpand\Dfootnote{D}%
1200                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1201     \global\advance\insert@count \@ne
1202   \else
1203     \Dfootnote{D}{\l@d@nums}{\@tag}{#1}%
1204   \fi\ignorespaces}

1205 \newcommand*{\Efootnote}[1]{%
1206   \ifnumberedpar@
```

```

1207 \xright@appenditem{\noexpand\vEfootnote{E}%
1208         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1209 \global\advance\insert@count \@ne
1210 \else
1211 \vEfootnote{E}{\l@d@nums}{\@tag}{#1}%
1212 \fi\ignorespaces}
1213

```

`\mpAfootins` For footnotes in minipages and the like, we need a new set of inserts.

```

\mpBfootins 1214 \newinsert\mpAfootins
\mpCfootins 1215 \newinsert\mpBfootins
\mpDfootins 1216 \newinsert\mpCfootins
\mpEfootins 1217 \newinsert\mpDfootins
1218 \newinsert\mpEfootins
1219

```

`\mpAfootnote` For footnotes in minipages and the like, we need a similar series of commands.

```

\mpBfootnote 1220 \newcommand*\mpAfootnote}[1]{%
\mpCfootnote 1221 \ifnumberedpar@
\mpDfootnote 1222 \xright@appenditem{\noexpand\mpvAfootnote{A}%
\mpEfootnote 1223         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1224 \global\advance\insert@count \@ne
1225 \else
1226 \mpvAfootnote{A}{\l@d@nums}{\@tag}{#1}%
1227 \fi\ignorespaces}
1228 \newcommand*\mpBfootnote}[1]{%
1229 \ifnumberedpar@
1230 \xright@appenditem{\noexpand\mpvBfootnote{B}%
1231         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1232 \global\advance\insert@count \@ne
1233 \else
1234 \mpvBfootnote{B}{\l@d@nums}{\@tag}{#1}%
1235 \fi\ignorespaces}
1236 \newcommand*\mpCfootnote}[1]{%
1237 \ifnumberedpar@
1238 \xright@appenditem{\noexpand\mpvCfootnote{C}%
1239         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1240 \global\advance\insert@count \@ne
1241 \else
1242 \mpvCfootnote{C}{\l@d@nums}{\@tag}{#1}%
1243 \fi\ignorespaces}
1244 \newcommand*\mpDfootnote}[1]{%
1245 \ifnumberedpar@
1246 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1247         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1248 \global\advance\insert@count \@ne
1249 \else
1250 \mpvDfootnote{D}{\l@d@nums}{\@tag}{#1}%
1251 \fi\ignorespaces}

```

```

1252 \newcommand*{\mpEfootnote}[1]{%
1253   \ifnumberedpar@
1254     \xright@appenditem{\noexpand\mpvEfootnote{E}%
1255       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1256     \global\advance\insert@count \@ne
1257   \else
1258     \mpvEfootnote{E}{\l@0|0|0|0|0|0}{#1}%
1259   \fi\ignorespaces}

```

22.3 Normal footnote formatting

The processing of each note is done by four principal macros: the `\vfootnote` macro takes the text of the footnote and does the `\insert`; it calls on the `\footfmt` macro to select the right fonts, print the line number and lemma, and do any other formatting needed for that individual note. Within the output routine, the two other macros, `\footstart` and `\footgroup`, are called; the first prints extra vertical space and a footnote rule, if desired; the second does any reformatting of the whole set of footnotes in this series for this page—such as paragraphing or division into columns—and then sends them to the page.

These four macros, and the other macros and parameters shown here, are distinguished by the ‘series letter’ that indicates which set of footnotes we’re dealing with—A, B, C, D, or E. The series letter always precedes the string `foot` in macro and parameter names. Hence, for the A series, the four macros are called `\vAfootnote`, `\Afootfmt`, `\Afootstart`, and `\Afootgroup`.

`\normalvfootnote` We now begin a series of commands that do ‘normal’ footnote formatting: a format much like that implemented in PLAIN `TEX`, in which each footnote is a separate paragraph.

`\normalvfootnote` takes the series letter as `#1`, and the entire text of the footnote is `#2`. It does the `\insert` for this note, calling on the `\footfmt` macro for this note series to format the text of the note.

```

1260 \newcommand*{\normalvfootnote}[2]{%
1261   \insert\csname #1footins\endcsname\bgroup
1262   \notefontsetup
1263   \footsplitskips
1264   \spaceskip=\z@skip \xspaceskip=\z@skip
1265   \csname #1footfmt\endcsname #2\egroup}

```

`\footsplitskips` Some setup code that is common for a variety of footnotes.

```

1266 \newcommand*{\footsplitskips}{%
1267   \interlinepenalty=\interfootnotelinepenalty
1268   \floatingpenalty=\@MM
1269   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
1270   \leftskip=\z@skip \rightskip=\z@skip}
1271

```

`\mpnormalvfootnote` And a somewhat different version for minipages.

```

1272 \newcommand*{\mpnormalvfootnote}[2]{%

```

```

1273 \global\setbox\@nameuse{mp#1footins}\vbox{%
1274   \unvbox\@nameuse{mp#1footins}
1275   \notefontsetup
1276   \hsize\columnwidth
1277   \@parboxrestore
1278   \color@begingroup
1279   \csname #1footfmt\endcsname #2\color@endgroup}}
1280

```

\ledsetnormalparstuff **\normalfootfmt** is a ‘normal’ macro to take the footnote line and page number information (see p. 55), and the desired text, and output what’s to be printed. Argument #1 contains the line and page number information and lemma font specifier; #2 is the lemma; #3 is the note’s text. This version is very rudimentary—it uses **\printlines** to print just the range of line numbers, followed by a square bracket, the lemma, and the note text; it’s intended to be copied and modified as necessary.

\par should always be redefined to **\endgraf** within the format macro (this is what **\normal@pars** does), to override any tricky stuff which might be done in the main text to get the lines numbered automatically (as set up by **\autopar**, for example).

```

1281 \newcommand*{\ledsetnormalparstuff}{%
1282   \normal@pars
1283   \parindent \z@ \parfillskip \z@ \@plus 1fil}
1284 \newcommand*{\normalfootfmt}[3]{%
1285   \ledsetnormalparstuff
1286   {\notenumfont\printlines#1}\strut\enspace
1287   {\select@lemmafont#1|#2}\rbracket\enskip#3\strut\par}
1288

```

\endashchar The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

The **\endashchar** macro is simply an en-dash from the normal font and is immune to changes in the surrounding font. The same goes for the full stop. These two are used in **\printlines**. The right bracket macro is the same again; it crops up in **\normalfootfmt** and the other footnote macros for controlling the format of footnotes.

```

1289 \def\endashchar{\textnormal{--}}
1290 \newcommand*{\fullstop}{\textnormal{.}}
1291 \newcommand*{\rbracket}{\textnormal{\thinspace]]}}
1292

```

The **\printlines** macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in **\l@d@nums**, in the form described on page 55:

the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

The original EDMAC code used several counters at this point, saying:

To simplify the logic, we use a lot of counters to tell us which numbers need to get printed (using 1 for yes, 0 for no, so that `\ifodd` tests for ‘yes’). The counter assignments are:

- `\@pnum` for page numbers;
- `\@ssub` for starting sub-line;
- `\@elin` for ending line;
- `\@esl` for ending sub-line; and
- `\@dash` for the dash between the starting and ending groups.

There’s no counter for the line number because it’s always printed.

LaTeX tends to use a lot of counters and packages should try and minimise the number of new ones they create. In line with this I have reverted to traditional booleans.

```

\ifl@d@pnum
\ifl@d@ssub 1293 \newif\ifl@d@pnum
\ifl@d@elin 1294 \l@d@pnumfalse
\ifl@d@esl 1295 \newif\ifl@d@ssub
\ifl@d@dash 1296 \l@d@ssubfalse
1297 \newif\ifl@d@elin
1298 \l@d@elinfalse
1299 \newif\ifl@d@esl
1300 \l@d@eslfalse
1301 \newif\ifl@d@dash
1302 \l@d@dashfalse

\ifledplinenum Sometimes it could be useful not to print the line number, or give it a symbolic value
\symplinenum (perhaps if there are several notes from the same line).
1303 \newif\ifledplinenum
1304 \ledplinenumtrue
1305 \newcommand*{\symplinenum}{}
1306

\l@dparsefootspec \l@dparsefootspec{<spec>}{<lemma>}{<text>} parses a footnote specification.
\l@dp@rsefootspec <lemma> and <text> are the lemma and text respectively. <spec> is the line and
\l@dparsedstartpage page number and lemma font specifier in \l@d@nums style format. The real work
\l@dparsedstartline is done by \l@dp@rsefootspec which defines macros holding the numeric values.
\l@dparsedstartsub 1307 \newcommand*{\l@dparsefootspec}[3]{\l@dp@rsefootspec#1|}
\l@dparsedendpage 1308 \def\l@dp@rsefootspec#1|#2|#3|#4|#5|#6|#7|{%
\l@dparsedendline 1309 \gdef\l@dparsedstartpage{#1}%
\l@dparsedendsub 1310 \gdef\l@dparsedstartline{#2}%
1311 \gdef\l@dparsedstartsub{#3}%
1312 \gdef\l@dparsedendpage{#4}%
1313 \gdef\l@dparsedendline{#5}%
1314 \gdef\l@dparsedendsub{#6}%
1315 }
```


Initialise the several number value macros.

```

1316 \def\l@dparsedstartpage{0}%
1317 \def\l@dparsedstartline{0}%
1318 \def\l@dparsedstartsub{0}%
1319 \def\l@dparsedendpage{0}%
1320 \def\l@dparsedendline{0}%
1321 \def\l@dparsedendsub{0}%
1322

```

\setprintlines First of all, we print the page numbers only if: 1) we're doing the lineation by page, and 2) the ending page number is different from the starting page number.

Just a reminder of the arguments:

```

\printlines    #1      | #2 | #3      | #4      | #5 | #6      | #7
\printlines start-page | line | subline | end-page | line | subline | font

```

The macro **\setprintlines** does the work of deciding what numbers should be printed. Its arguments are the same as the first 6 of **\printlines**.

```

1323 \newcommand*{\setprintlines}[6]{%
1324   \l@dpnumfalse \l@ddashfalse
1325   \ifbypage@
1326     \ifnum#4=#1 \else
1327       \l@dpnumtrue
1328       \l@ddashtrue
1329     \fi
1330   \fi

```

We print the ending line number if: (1) we're printing the ending page number, or (2) it's different from the starting line number.

```

1331   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
1332   \ifnum#2=#5 \else
1333     \l@d@elintrue
1334     \l@ddashtrue
1335   \fi

```

We print the starting sub-line if it's nonzero.

```

1336   \l@d@ssubfalse
1337   \ifnum#3=0 \else
1338     \l@d@ssubtrue
1339   \fi

```

We print the ending sub-line if it's nonzero and: (1) it's different from the starting sub-line number, or (2) the ending line number is being printed.

```

1340   \l@d@eslfalse
1341   \ifnum#6=0 \else
1342     \ifnum#6=#3
1343       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
1344     \else
1345       \l@d@esltrue
1346       \l@ddashtrue
1347     \fi
1348   \fi}

```

`\printlines` Now we're ready to print it all. If the lineation is by `pstart`, we print the `pstart`.

```

1349 \def\printlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
1350   \ifbypstart%
1351   \ifl@dpairing%
1352     \ifledRcol%
1353       \thepstartR%
1354     \else%
1355       \thepstartL%
1356     \fi%
1357   \else%
1358     \thepstart%
1359   \fi%
1360 \fi%
1361 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%

```

One subtlety left here is when to print a period between numbers. But the only instance in which this is tricky is for the ending sub-line number: it could be coming after the starting sub-line number (in which case we want only the dash) or after an ending line number (in which case we need to insert a period).

```

1362 \ifl@d@pnum #1\fullstop\fi

```

The other thing is whether to print the real starting line number or a symbolic value.

```

1363 \ifledplinenum \linenumrep{#2}\else \symplinenum\fi
1364 \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
1365 \ifl@d@dash \endashchar\fi
1366 \ifl@d@pnum #4\fullstop\fi
1367 \ifl@d@elin \linenumrep{#5}\fi
1368 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi
1369 \endgroup}
1370

```

`\normalfootstart` `\normalfootstart` is a standard footnote-starting macro, called in the output routine whenever there are footnotes of this series to be printed: it skips a bit and then draws a rule.

Any `footstart` macro must put onto the page something that takes up space exactly equal to the `\skip\footins` value for the associated series of notes. `TEX` makes page computations based on that `\skip` value, and the output pages will suffer from spacing problems if what you add takes up a different amount of space.

The `\leftskip` and `\rightskip` values are both zeroed here. Similarly, these skips are cancelled in the `vfootnote` macros for the various types of notes. Strictly speaking, this is necessary only if you are using paragraphed footnotes, but we have put it here and in the other `vfootnote` macros too so that the behavior of `ledmac` in this respect is general across all footnote types (you can change this). What this means is that any `\leftskip` and `\rightskip` you specify applies to the main text, but not the footnotes. The footnotes continue to be of width `\hsize`.

```

1371 \newcommand*{\normalfootstart}[1]{%
1372   \vskip\skip\csname #1footins\endcsname
1373   \leftskip0pt \rightskip0pt
1374   \csname #1footnoterule\endcsname}

```

`\normalfootnoterule` `\normalfootnoterule` is a standard footnote-rule macro, for use by a `footstart` macro: just the same as the PLAIN \TeX footnote rule.

```
1375 \let\normalfootnoterule=\footnoterule
```

`\normalfootgroup` `\normalfootgroup` is a standard footnote-grouping macro: it sends the contents of the footnote-insert box to the output page without alteration.

```
1376 \newcommand*{\normalfootgroup}[1]{\unvbox\csize #1footins\endcsize}
1377
```

`\mpnormalfootgroup` A somewhat different version for minipages.

```
1378 \newcommand*{\mpnormalfootgroup}[1]{\{
1379   \vskip\skip\@nameuse{mp#1footins}
1380   \normalcolor
1381   \@nameuse{#1footnoterule}
1382   \unvbox\csize mp#1footins\endcsize}\}
1383
```

22.4 Standard footnote definitions

`\footnormal` We can now define all the parameters for the five series of footnotes; initially they use the ‘normal’ footnote formatting, which is set up by calling `\footnormal`. You can switch to another type of formatting by using `\footparagraph`, `\foottwocol`, or `\footthreecol`.

Switching to a variation of ‘normal’ formatting requires changing the quantities defined in `\footnormal`. The best way to proceed would be to make a copy of this macro, with a different name, make your desired changes in that copy, and then invoke it, giving it the letter of the footnote series you wish to control.

(We have not defined baseline skip values like `\baselineskip`, since this is one of the quantities set in `\notefontsetup`.)

What we want to do here is to say something like the following for each footnote series. (This is an example, not part of the actual `ledmac` code.)

```
\newinsert\Afootins
\skip\Afootins=12pt plus5pt minus5pt
\count\Afootins=1000
\dimen\Afootins=0.8\vsiz
\let\Afootnote=\normalvfootnote \let\Afootfmt=\normalfootfmt
\let\Afootstart=\normalfootstart \let\Afootgroup=\normalfootgroup
\let\Afootnoterule=\normalfootnoterule
```

Instead of repeating ourselves, we define a `\footnormal` macro that makes all these assignments for us, for any given series letter. This also makes it easy to change from any different system of formatting back to the `normal` setting.

`\ledfootinsdim` Have a constant value for the `\dimen\footins`

```
1384 \newcommand*{\ledfootinsdim}{0.8\vsiz}
1385
```

We begin by defining the five new insertion classes, and some `count` registers; these are `\outer` operations that can't be done inside `\footnormal`.

```
1386 \newinsert\Afootins \newinsert\Bfootins
1387 \newinsert\Cfootins \newinsert\Dfootins
1388 \newinsert\Efootins
```

Now we set up the `\footnormal` macro itself. It takes one argument: the footnote series letter.

```
1389 \newcommand*{\footnormal}[1]{%
1390   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
1391   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
1392   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
1393   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
1394   \expandafter\let\csname #1footnoterule\endcsname=%
1395                                   \normalfootnoterule
1396   \count\csname #1footins\endcsname=1000
1397   \dimen\csname #1footins\endcsname=\ledfootinsdim
1398   \skip\csname #1footins\endcsname=1.2em \@plus .6em \@minus .6em
```

Now do the setup for minipage footnotes. We use as much as possible of the normal setup as we can (so the notes will have a similar layout).

```
1399   \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
1400   \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
1401   \count\csname mp#1footins\endcsname=1000
1402   \dimen\csname mp#1footins\endcsname=\ledfootinsdim
1403   \skip\csname mp#1footins\endcsname=1.2em \@plus .6em \@minus .6em
1404 }
1405
```

Some of these values deserve comment: the `\dimen` setting allows 80% of the page to be occupied by notes; the `\skip` setting is deliberately flexible, since pages with lots of notes attached to many of the lines can be a bit hard for T_EX to make.

And finally, we initialize the formatting for all the footnote series to be normal.

```
1406 \footnormal{A}
1407 \footnormal{B}
1408 \footnormal{C}
1409 \footnormal{D}
1410 \footnormal{E}
1411
```

22.5 Paragraphed footnotes

The `paragraphed-footnote` option reformats all the footnotes of one series for a page into a single paragraph; this is especially appropriate when the notes are numerous and brief. The code is based on *The T_EXbook*, pp.398–400, with alterations for our environment. This algorithm uses a considerable amount of save-stack space: a T_EX of ordinary size may not be able to handle more than about 100 notes of this kind on a page.

`\footparagraph` The `\footparagraph` macro sets up everything for one series of footnotes so that they'll be paragraphed; it takes the series letter as argument. We include the setting of `\count\footins` to 1000 for the footnote series just in case you are switching to paragraphed footnotes after having columnar ones, since they change this value (see below).

It is important to call `\footparagraph` only after `\hsize` has been set for the pages that use this series of notes; otherwise TeX will try to put too many or too few of these notes on each page. If you need to change the `\hsize` within the document, call `\footparagraph` again afterwards to take account of the new value. The argument of `\footparagraph` is the letter (A–E) denoting the series of notes to be paragraphed.

```

1412 \newcommand*{\footparagraph}[1]{%
1413   \expandafter\let\csname #1footstart\endcsname=\parafootstart
1414   \expandafter\let\csname v#1footnote\endcsname=\para@vfootnote
1415   \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
1416   \expandafter\let\csname #1footgroup\endcsname=\para@footgroup
1417   \count\csname #1footins\endcsname=1000
1418   \para@footsetup{#1}

```

And the extra setup for minipages.

```

1419   \expandafter\let\csname mpv#1footnote\endcsname=\mppara@vfootnote
1420   \expandafter\let\csname mp#1footgroup\endcsname=\mppara@footgroup
1421   \count\csname mp#1footins\endcsname=1000
1422 }
1423

```

You can redefine the `\parafootftmsep` command to print a separator between each paragraphed footnote (on the same page). A usual separator is a double pipe (`||`). To add double-pipe separators:

```
\renewcommand{\parafootftmsep}{\thinspace$||$\enspace}
```

`\footfudgefiddle` For paragraphed footnotes TeX has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say to 70) to increase the estimate.

```
1424 \providecommand{\footfudgefiddle}{64}
```

`\para@footsetup` `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hsize`. We assume that the proper value of `\baselineskip` for the footnotes (normally 9 pt) has been set already, in `\notefontsetup`. The argument of the macro is again the note series letter.

I think that `\columnwidth` should be used here for LaTeX, not `\hsize`. I've also included `\footfudgefiddle`.

```

1425 \newcommand*{\para@footsetup}[1]{\notefontsetup
1426   \dimen0=\baselineskip

```

```

1427 \multiply\dimen0 by 1024
1428 \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
1429 \expandafter
1430 \xdef\csname #1footfudgefactor\endcsname{%
1431   \expandafter\strip@pt\dimen0 }}}
1432

```

EDMAC defines `\en@number` which does the same as the LaTeX kernel `\strip@pt`, namely strip the characters `pt` from a `dimen` value. I'll use `\strip@pt`.

`\parafootstart` `\parafootstart` is the same as `\normalfootstart`, but we give it again to ensure that `\rightskip` and `\leftskip` are zeroed (this needs to be done before `\para@footgroup` in the output routine). You might have decided to change this for other kinds of note, but here it should stay as it is. The size of paragraphed notes is calculated using a fudge factor which in turn is based on `\hsize`. So the paragraph of notes needs to be that wide.

The argument of the macro is again the note series letter.

```

1433 \newcommand*{\parafootstart}[1]{%
1434   \rightskip=0pt \leftskip=0pt \parindent=0pt
1435   \vskip\skip\csname #1footins\endcsname
1436   \csname #1footnoterule\endcsname}

```

`\para@vfootnote` `\para@vfootnote` is a version of the `\vfootnote` command that's used for paragraphed notes. It gets appended to the `\inserts@list` list by an outer-level footnote command like `\Afootnote`. The first argument is the note series letter; the second is the full text of the printed note itself, including line numbers, lemmata, and footnote text.

The initial model for this insertion is, of course, the `\insert\footins` definition in *The TeXbook*, p. 398. There, the footnotes are first collected up in `hboxes`, and these `hboxes` are later unpacked and stuck together into a paragraph.

However, Michael Downes has pointed out that because text in `hboxes` gets typeset in restricted horizontal mode, there are some undesirable side-effects if you later want to break such text across lines. In restricted horizontal mode, where TeX does not expect to have to break lines, it does not insert certain items like `\discretionary`s. If you later unbox these `hboxes` and stick them together, as the *TeXbook* macros do to make these footnotes, you lose the ability to hyphenate after an explicit hyphen. This can lead to overfull `hboxes` when you would not expect to find them, and to the uninitiated it might be very hard to see why the problem had arisen.²⁴

Wayne Sullivan pointed out to us another subtle problem that arises from the same cause: TeX also leaves the `\language` whatsit nodes out of the horizontal list.²⁵ So changes from one language to another will not invoke the proper hyphenation rules in such footnotes. Since critical editions often do deal with several languages, especially in footnotes, we really ought to get this bit of code right.

²⁴Michael Downes, 'Line Breaking in `\unhboxed` Text', *TUGboat* **11** (1990), pp. 605–612.

²⁵See *The TeXbook*, p. 455 (editions after January 1990).

To get around these problems, Wayne suggested emendations to the *TeXbook* versions of these macros which are broadly the same as those described by Michael: the central idea (also suggested by Donald Knuth in a letter to Michael) is to avoid collecting the text in an `\hbox` in the first place, but instead to collect it in a `\vbox` whose width is (virtually) infinite. The text is therefore typeset in unrestricted horizontal mode, as a paragraph consisting of a single long line. Later, there is an extra level of unboxing to be done: we have to unpack the `\vbox`, as well as the `\hboxes` inside it, but that's not too hard. For details, we refer you to Michael's article, where the issues are clearly explained.²⁶ Michael's unboxing macro is called `\unvbx`: `unvbox`, extract the last line, and `unhbox` it.

Doing things this way has an important consequence: as Michael pointed out, you really can't put an explicit line-break into a note built in a `\vbox` the way we are doing.²⁷ In other words, be very careful not to say `\break`, or `\penalty-10000`, or any equivalent inside your para-footnote. If you do, most of the note will probably disappear. You *are* allowed to make strong suggestions; in fact `\penalty-9999` will be quite okay. Just don't make the break mandatory. We haven't applied any of Michael's solutions here, since we feel that the problem is exiguous, and `ledmac` is quite baroque enough already. If you think you are having this problem, look up Michael's solutions.

One more thing; we set `\leftskip` and `\rightskip` to zero. This has the effect of neutralizing any such skips which may apply to the main text (cf. p. 98 above). We need to do this, since `footfudgefactor` is calculated on the assumption that the notes are `\hsize` wide.

So, finally, here is the modified foot-paragraph code, which sets the footnote in vertical mode so that language and discretionary nodes are included.

```

1437 \newcommand*{\para@vfootnote}[2]{%
1438   \insert\csname #1footins\endcsname
1439   \bgroup
1440     \notefontsetup
1441     \footplitskips
1442     \setbox0=\vbox{\hsize=\maxdimen
1443       \noindent\csname #1footfmt\endcsname#2}%
1444     \setbox0=\hbox{\unvbx0}%
1445     \dp0=0pt
1446     \ht0=\csname #1footfudgefactor\endcsname\wd0

```

Here we produce the contents of the footnote from box 0, and add a penalty of 0 between boxes in this insert.

```

1447   \box0
1448   \penalty0
1449 \egroup}
1450

```

The final penalty of 0 was added here at Wayne's suggestion to avoid a weird page-breaking problem, which occurs on those occasions when `TeX` attempts to

²⁶Wayne supplied his own macros to do this, but since they were almost identical to Michael's, we have used the latter's `\unvbx` macro since it is publicly documented.

²⁷'Line Breaking', p. 610.

split foot paragraphs. After trying out such a split (see *The TeXbook*, p.124), T_EX inserts a penalty of -10000 here, which nearly always forces the break at the end of the whole footnote paragraph (since individual notes can't be split) even when this leads to an overfull vbox. The change above results in a penalty of 0 instead which allows, but doesn't force, such breaks. This penalty of 0 is later removed, after page breaks have been decided, by the `\unpenalty` macro in `\makehboxofhboxes`. So it does not affect how the footnote paragraphs are typeset (the notes still have a penalty of -10 between them, which is added by `\parafootfmt`).

`\mppara@vfootnote` This version is for minipages.

```

1451 \newcommand*{\mppara@vfootnote}[2]{%
1452   \global\setbox\@nameuse{mp#1footins}\vbox{%
1453     \unvbox\@nameuse{mp#1footins}%
1454     \notefontsetup
1455     \footssplitsskip
1456     \setbox0=\vbox{\hsize=\maxdimen
1457       \noindent\color@begingroup\csname #1footfmt\endcsname #2\color@endgroup}%
1458     \setbox0=\hbox{\unvxh0}%
1459     \dp0=\z@
1460     \ht0=\csname #1footfudgefactor\endcsname\wd0
1461     \box0
1462     \penalty0
1463   }}
1464
```

`\unvxh` Here is Michael's definition of `\unvxh`, used above. Michael's macro also takes care to remove some unwanted penalties and glue that T_EX automatically attaches to the end of paragraphs. When T_EX finishes a paragraph, it throws away any remaining glue, and then tacks on the following items: a `\penalty` of 10000, a `\parfillskip` and a `\rightskip` (*The TeXbook*, pp.99–100). `\unvxh` cancels these unwanted paragraph-final items using `\unskip` and `\unpenalty`.

```

1465 \newcommand*{\unvxh}[1]{%
1466   \setbox0=\vbox{\unvbox#1%
1467     \global\setbox1=\lastbox}%
1468   \unhbox1
1469   \unskip           % remove \rightskip,
1470   \unskip           % remove \parfillskip,
1471   \unpenalty        % remove \penalty of 10000,
1472   \hskip\ipn@skip}  % but add the glue to go between the notes
1473
```

`\interparanoteglue` Close observers will notice that we snuck some glue called `\ipn@skip` onto the end of the hbox produced by `\unvxh` in the above macro.

We want to be able to have some glue between our paragraphed footnotes. But since we are initially setting our notes in internal vertical mode, as little paragraphs, any paragraph-final glue will get discarded. Since `\unvxh` is already

busy fiddling with glue and penalties at the end of these paragraphs, we take advantage of the opportunity to provide our inter-note spacing.

We collect the value of the inter-parafootnote glue value as the parameter of a macro called—wait for it—`\interparanoteglue`. We put this value into the value of a glue register `\ipn@skip` (inter-para-note-skip) making sure first to set the current font to the value normally used in footnotes so that the value of an `em` will be taken from the right font.

```

1474 \newskip\ipn@skip
1475 \newcommand*{\interparanoteglue}[1]{%
1476     {\notefontsetup\global\ipn@skip=#1 \relax}}
1477 \interparanoteglue{1em plus.4em minus.4em}
1478

```

There is a point to be careful about regarding the `\interparanoteglue`. Remember that in `\para@vfootnote` we do some measurements on the footnote box, and use the resulting size to make an estimate of how much the note will contribute to the height of our final footnote paragraph. This information is used by the output routine to allocate the right amount of vertical space on the page for the notes (*The TeXbook*, pp. 398–399).

The length of the footnote includes the natural size of the glue specified by `\interparanoteglue`, but not its stretch or shrink components, since at this point the note has no need to stretch or shrink. Later, when the paragraph is actually composed by `\parafootgroup` in the output routine, \TeX will almost certainly do some stretching and shrinking of this glue in order to make the paragraph look nice. Probably the stretching and shrinking over the whole paragraph will cancel each other out. But if not, the actual vertical size of the paragraph may not match the size the output routine had been told to expect, and you may get an overfull/underfull `\vbox` message from the output routine. To minimize the risk of this, you can do two things: keep the `plus` and `minus` components of `\interparanoteglue` small compared with its natural glue, and keep them the same as each other. As a general precaution, keep the size and flexibility of the `\skip\footins` glue on the high side too: because the reckoning is approximate, footnote blocks may be up to a line bigger or smaller than the output routine allows for, so keep some flexible space between the text and the notes.

`\parafootfmt` `\parafootfmt` is `\normalfootfmt` adapted to do the special stuff needed for paraphed notes—leaving out the `\endgraf` at the end, sticking in special penalties and kern, and leaving out the `\footstrut`. The first argument is the line and page number information, the second is the lemma, and the third is the text of the footnote.

```

1479 \newcommand*{\parafootfmt}[3]{%
1480     \insertparafootfmtsep%
1481     \ledsetnormalparstuff
1482     {\notenumfont\printlines#1|}\enspace
1483     {\select@lemmafont#1|#2}\rbracket\enskip
1484     #3\penalty-10 }

```

Note that in the above definition, the penalty of `-10` encourages a line break

between notes, so that notes have a slight tendency to begin on new lines. The `\insertparafootftmsep` command is used to insert the `\parafootftmsep` between each note in the *same* page.

`\para@footgroup` This `footgroup` code is modelled on the macros in *The TeXbook*, p. 399. The only difference is the `\unpenalty` in `\makehboxofhboxes`, which is there to remove the penalty of 0 which was added to the end of each footnote by `\para@vfootnote`.

The call to `\notefontsetup` is to ensure that the correct `\baselineskip` for the footnotes is used. The argument is the note series letter.

```
1485 \newcommand*{\para@footgroup}[1]{%
1486   \unvbox\csname #1footins\endcsname
1487   \makehboxofhboxes
1488   \setbox0=\hbox{\unhbox0 \removehboxes}%
1489   \notefontsetup
1490   \noindent\unhbox0\par}
1491
```

`\mppara@footgroup` The minipage version.

```
1492 \newcommand*{\mppara@footgroup}[1]{%
1493   \vskip\skip\@nameuse{mp#1footins}
1494   \normalcolor
1495   \@nameuse{#1footnoterule}%
1496   \unvbox\csname mp#1footins\endcsname
1497   \makehboxofhboxes
1498   \setbox0=\hbox{\unhbox0 \removehboxes}%
1499   \notefontsetup
1500   \noindent\unhbox0\par}}
1501
```

`\makehboxofhboxes`

```
\removehboxes 1502 \newcommand*{\makehboxofhboxes}{\setbox0=\hbox{}%
1503   \loop
1504     \unpenalty
1505     \setbox2=\lastbox
1506     \ifhbox2
1507       \setbox0=\hbox{\box2\unhbox0}%
1508     \repeat}
1509
1510 \newcommand*{\removehboxes}{\setbox0=\lastbox
1511   \ifhbox0{\removehboxes}\unhbox0 \fi}
1512
```

22.5.1 Insertion of footnotes separator

`\parafootftmsep` The `\parafootftmsep` macro is inserted between each paragraphed footnote. The default value is empty, but the user can redefine it via `\renewcommand`.

```
1513 \newcommand{\parafootftmsep}{}
```

The command `\insertparafootftmsep` must be called at the beginning of `\parafootftm` (and like commands). `\insertparafootftmsep` checks to see if the page number has changed since the previous note. If not, `\insertparafootftmsep` calls `\parafootftmsep`.

```
\prevpage@num
\insertparafootftmsep 1514 \newcount\prevpage@num
1515 \newcommand{\insertparafootftmsep}{%
1516     \ifnum\prevpage@num=\page@num%
1517         \parafootftmsep%
1518     \fi%
1519     \global\prevpage@num=\page@num%
1520 }
```

22.6 Columnar footnotes

`\rigidbalance` We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\rigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the syntax of the arguments so that they don't depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

The LaTeX `\line` macro has no relationship to the TeX `\line`. The LaTeX equivalent is `\@@line`.

```
1521 \newcount\@k \newdimen\@h
1522 \newcommand*\rigidbalance{[3]{\setbox0=\box#1 \@k=#2 \@h=#3
1523     \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
1524     \valign{##\vfil\cr\dosplits}}}}
1525
1526 \newcommand*\dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
1527     \global\advance\@k-1\cr\dosplits\fi}
1528
1529 \newcommand*\splitoff{\dimen0=\ht0
1530     \divide\dimen0 by\@k \advance\dimen0 by\@h
1531     \setbox2 \vsplit0 to \dimen0
1532     \unvbox2 }
1533
```

Three columns

`\footthreecol` You say `\footthreecol{A}` to have the A series of footnotes typeset in three columns. It is important to call this only after `\hsize` has been set for the document.

```
1534 \newcommand*\footthreecol{[1]{%
1535     \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
1536     \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
1537     \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
```

```
1538 \threecolfootsetup{#1}
```

The additional setup for minipages.

```
1539 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
1540 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
1541 \mpthreecolfootsetup{#1}
1542 }
1543
```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (p. 98 above).

`\threecolfootsetup` The `\threecolfootsetup` macro calculates and sets some numbers for three-column footnotes.

We set the `\count` of the foot insert to 333. Each footnote can be thought of as contributing only one third of its height to the page, since the footnote insertion has been made as a long narrow column, which then gets trisected by the `\rigidbalance` routine (inside `\threecolfootgroup`). These new, shorter columns are saved in a box, and then that box is *put back* into the footnote insert, replacing the original collection of footnotes. This new box is, therefore, only about a third of the height of the original one.

The `\dimen` value for this note series has to change in the inverse way: it needs to be three times the actual limit on the amount of space these notes are allowed to fill on the page, because when T_EX is accumulating material for the page and checking that limit, it doesn't apply the `\count` scaling.

```
1544 \newcommand*{\threecolfootsetup}[1]{%
1545   \count\csname #1footins\endcsname 333
1546   \multiply\dimen\csname #1footins\endcsname \thr@@}
```

`\mpthreecolfootsetup` The setup for minipages.

```
1547 \newcommand*{\mpthreecolfootsetup}[1]{%
1548   \count\csname mp#1footins\endcsname 333
1549   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
1550
```

`\threecolvfootnote` `\threecolvfootnote` is the `\vfootnote` command for three-column notes. The call to `\notefontsetup` ensures that the `\splittopskip` and `\splitmaxdepth` take their values from the right `\strutbox`: the one used in footnotes. Note especially the importance of temporarily reducing the `\hsize` to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal `\hsize` is, say, 10 cm, then each column will be $0.3 \times 10 = 3$ cm wide, leaving a gap of 1 cm spread equally between columns (i.e., .5 cm between each).

The arguments are 1) the note series letter and 2) the full text of the note (including numbers, lemma and text).

```
1551 \newcommand*{\threecolvfootnote}[2]{%
1552   \insert\csname #1footins\endcsname\bgroup
1553   \notefontsetup
1554   \footsplittskips
1555   \csname #1footfmt\endcsname #2\egroup}
```

`\threecolfootfmt` `\threecolfootfmt` is the command that formats one note. It uses `\raggedright`, which will usually be preferable with such short lines. Setting the `\parindent` to zero means that, within each individual note, the lines begin flush left.

The arguments are 1) the line numbers, 2) the lemma and 3) the text of the `-footnote` command.

```

1556 \newcommand*{\threecolfootfmt}[3]{%
1557   \normal@pars
1558   \hsize .3\hsize
1559   \parindent=0pt
1560   \tolerance=5000
1561   \raggedright
1562   \leavevmode
1563   \strut{\notenumfont\printlines#1}\enspace
1564   {\select@lemmafnt#1|#2}\rbracket\enskip
1565   #3\strut\par\allowbreak}

```

`\threecolfootgroup` And here is the `footgroup` macro that's called within the output routine to re-group the notes into three columns. Once again, the call to `\notefontsetup` is there to ensure that it is the right `\splittopskip`—the one used in footnotes—which is used to provide the third argument for `\rigidbalance`. This third argument (`\@h`) is the `topskip` for the box containing the text of the footnotes, and does the job of making sure the top lines of the columns line up horizontally. In *The TeXbook*, p.398, Donald Knuth suggests retrieving the output of `\rigidbalance`, putting it back into the insertion box, and then printing the box. Here, we just print the `\line` which comes out of `\rigidbalance` directly, without any re-boxing.

```

1566 \newcommand*{\threecolfootgroup}[1]{\notefontsetup
1567   \splittopskip=\ht\strutbox
1568   \expandafter
1569   \rigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}}

```

`\mpthreecolfootgroup` The setup for minipages.

```

1570 \newcommand*{\mpthreecolfootgroup}[1]{%
1571   \vskip\skip\@nameuse{mp#1footins}
1572   \normalcolor
1573   \@nameuse{#1footnoterule}
1574   \splittopskip=\ht\strutbox
1575   \expandafter
1576   \rigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
1577

```

Two columns

`\foottwocol` You say `\foottwocol{A}` to have the A series of footnotes typeset in two columns. It is important to call this only after `\hsize` has been set for the document.

```

1578 \newcommand*{\foottwocol}[1]{%
1579   \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
1580   \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt

```

```

1581 \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
1582 \twocolfootsetup{#1}

```

The additional setup for minipages.

```

1583 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
1584 \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
1585 \mptwocolfootsetup{#1}
1586 }
1587

```

`\twocolfootsetup` Here is a series of macros which are very similar to their three-column counterparts.
`\twocolvfootnote` In this case, each note is assumed to contribute only a half a line of text. And
`\twocolfootfmt` the notes are set in columns `0.45\hsize` wide, giving a gap between them of one
`\twocolfootgroup` tenth of the `\hsize`.

```

1588 \newcommand*{\twocolfootsetup}[1]{%
1589   \count\csname #1footins\endcsname 500
1590   \multiply\dimen\csname #1footins\endcsname \tw@}
1591 \newcommand*{\twocolvfootnote}[2]{\insert\csname #1footins\endcsname\bgroup
1592   \notefontsetup
1593   \footssplitsskip
1594   \csname #1footfmt\endcsname #2\egroup}
1595 \newcommand*{\twocolfootfmt}[3]{%
1596   \normal@pars
1597   \hsize .45\hsize
1598   \parindent=0pt
1599   \tolerance=5000
1600   \raggedright
1601   \leavevmode
1602   \strut{\notenumfont\printlines#1|}\enspace
1603   {\select@lemmafnt#1|#2}\rbracket\enskip
1604   #3\strut\par\allowbreak}
1605 \newcommand*{\twocolfootgroup}[1]{\notefontsetup
1606   \splittopskip=\ht\strutbox
1607   \expandafter
1608   \rigidbalance\csname #1footins\endcsname \tw@ \splittopskip}}
1609

```

`\mptwocolfootsetup` The versions for minipages.

```

\mptwocolfootgroup 1610 \newcommand*{\mptwocolfootsetup}[1]{%
1611   \count\csname mp#1footins\endcsname 500
1612   \multiply\dimen\csname mp#1footins\endcsname \tw@}
1613 \newcommand*{\mptwocolfootgroup}[1]{%
1614   \vskip\skip\@nameuse{mp#1footins}
1615   \normalcolor
1616   \@nameuse{#1footnoterule}
1617   \splittopskip=\ht\strutbox
1618   \expandafter
1619   \rigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
1620

```

23 Output routine

Now we begin the output routine and associated things.

I have deleted all the crop mark code.

There are a couple of macros from plain TeX that we need (at least for now).

```

\pageno    \pageno is a page number, starting at 1, and \advancepageno increments the num-
\advancepageno ber.
1621 \countdef\pageno=0 \pageno=1
1622 \newcommand*\advancepageno*{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
1623   \else\global\advance\pageno\@ne\fi}
1624
```

The next portion is probably the trickiest part of moving from TeX to LaTeX. The original code is below, but we need something very different.

This is a new output routine, with changes to handle printing all our footnotes. Those changes have not been added directly, but are in macros that get called here: that should make it easier to see what would need to be taken over to a different output routine. We continue to use the `\pagebody`, `\makeheadline`, `\makefootline`, and `\dosupereject` macros of PLAIN TeX; for those macros, and the original version of `\output`, see *The TeXbook*, p.364.

```

\output{\edmac@output}
\def\edmac@output{\shipout\vbox{\normal@pars
  \vbox{\makeheadline\pagebody\makefootline}%
}%
\advancepageno
\ifnum\outputpenalty>-\@MM\else\dosupereject\fi}

\def\pagecontents{\page@start
\ifvoid\topins\else\unvbox\topins\fi
\dimen@=\dp\cc1v \unvbox\cc1v % open up \box255
\do@feet
\ifr@ggedbottom \kern-\dimen@ \vfil \fi}

```

`\do@feet` ships out all the footnotes. Standard EDMAC has only five feet, but there is nothing in principal to prevent you from creating an arachnoid or centipedal edition; straightforward modifications of EDMAC are all that's required. However, the myriapedal edition is ruled out by TeX's limitations: the number of insertion classes is limited to 255.

```

\def\do@feet{%
\ifvoid\footins\else
  \vskip\skip\footins
  \footnoterule
  \unvbox\footins
\fi
\ifvoid\Afootins\else

```

```

\Afootstart{A}\Afootgroup{A}%
\fi
\ifvoid\Bfootins\else
\Bfootstart{B}\Bfootgroup{B}%
\fi
\ifvoid\Cfootins\else
\Cfootstart{C}\Cfootgroup{C}%
\fi
\ifvoid\Dfootins\else
\Dfootstart{D}\Dfootgroup{D}%
\fi
\ifvoid\Efootins\else
\Efootstart{E}\Efootgroup{E}%
\fi}

```

For information (and so that I don't forget it), the code that now follows is part of the standard LaTeX output routine.

With luck we might only have to change `\@makecol` and `\@reinserts`. The kernel definition of these, and perhaps some other things, is:

```

\gdef \@makecol {%
\ifvoid\footins
\setbox\@outputbox \box\@cclv
\else
\setbox\@outputbox \vbox {%
\boxmaxdepth \@maxdepth
\@tempdima\dp\@cclv
\unvbox \@cclv
\vskip \skip\footins
\color@begingroup
\normalcolor
\footnoterule
\unvbox \footins
\color@endgroup
}%
\fi
\xdef\@freelist{\@freelist\@midlist}%
\global \let \@midlist \@empty
\@combinefloats
\ifvbox\@kludgeins
\@makespecialcolbox
\else
\setbox\@outputbox \vbox to\@colht {%
\@texttop
\dimen@ \dp\@outputbox
\unvbox\@outputbox
\vskip -\dimen@
\@textbottom
}%

```



```

\fi
\global \maxdepth \@maxdepth
}

\gdef \@reinserts{%
  \ifvoid\footins\else\insert\footins{\unvbox\footins}\fi
  \ifvbox\@kludgeins\insert\@kludgeins{\unvbox\@kludgeins}\fi
}

```

Now we start actually changing things.

`\m@m@makecolfloats` These macros are defined in the memoir class and form part of the definition of `\m@m@makecoltext` `\@makecol`.

```

\m@m@makecolintro 1625 \providecommand{\m@m@makecolfloats}{%
1626   \xdef\@freelist{\@freelist\@midlist}%
1627   \global \let \@midlist \@empty
1628   \@combinefloats}
1629 \providecommand{\m@m@makecoltext}{%
1630   \ifvbox\@kludgeins
1631     \@makespecialcolbox
1632   \else
1633     \setbox\@outputbox \vbox to\@colht {%
1634       \@texttop
1635       \dimen@ \dp\@outputbox
1636       \unvbox\@outputbox
1637       \vskip -\dimen@
1638       \@textbottom}%
1639   \fi}
1640 \providecommand{\m@m@makecolintro}{%
1641

```

`\l@d@makecol` This is a partitioned version of the ‘standard’ `\@makecol`, with the initial code put into another macro.

```

1642 \gdef\l@d@makecol{%
1643   \l@ddofootinsert
1644   \m@m@makecolfloats
1645   \m@m@makecoltext
1646   \global \maxdepth \@maxdepth}
1647

```

`\l@ddofootinsert` This macro essentially holds the initial portion of the kernel `\@makecol` code.

```

1648 \newcommand*\l@ddofootinsert{%
1649   %% \page@start
1650   \ifvoid\footins
1651     \setbox\@outputbox \box\@ccclv
1652   \else
1653     \setbox\@outputbox \vbox {%
1654       \boxmaxdepth \@maxdepth

```

```

1655      \@tempdima\dp\@cclv
1656      \unvbox \@cclv
1657      \vskip \skip\footins
1658      \color@begingroup
1659      \normalcolor
1660      \footnoterule
1661      \unvbox \footins
1662      \color@endgroup
1663      }%
1664      \fi

```

That's the end of the copy of the kernel code. We finally call a macro to handle all the additional EDMAC feet.

```

1665      \l@ddoxtrafeet
1666  }
1667

```

`\doxtrafeet` `\doxtrafeet` is the code extending `\@makecol` to cater for the extra ledmac feet. We have two classes of extra footnotes. We order the footnote inserts so that the regular footnotes are first, then class 1 (familiar footnotes) and finally class 2 (critical footnotes).

```

1668 \newcommand*{\l@ddoxtrafeet}{%
1669   \doxtrafeeti
1670   \doxtrafeetii}
1671

```

`\doxtrafeetii` `\doxtrafeetii` is the code extending `\@makecol` to cater for the extra critical feet (class 2 feet). NOTE: the code is likely to be 'featurefull'.

```

1672 \newcommand*{\doxtrafeetii}{%
1673   \setbox\@outputbox \vbox{%
1674     \unvbox\@outputbox
1675     \@opxtrafeetii}}

```

`\@opxtrafeetii` The extra critical feet to be added to the output.

```

1676 \newcommand*{\@opxtrafeetii}{%
1677   \ifvoid\Afootins\else\Afootstart{A}\Afootgroup{A}\fi
1678   \ifvoid\Bfootins\else\Bfootstart{B}\Bfootgroup{B}\fi
1679   \ifvoid\Cfootins\else\Cfootstart{C}\Cfootgroup{C}\fi
1680   \ifvoid\Dfootins\else\Dfootstart{D}\Dfootgroup{D}\fi
1681   \ifvoid\Efootins\else\Efootstart{E}\Efootgroup{E}\fi}
1682

```

`\l@ddodoreinxtrafeet` `\l@ddodoreinxtrafeet` is the code for catering for the extra footnotes within `\@reinserts`. The implementation may well have to change. We use the same classes and ordering as in `\l@ddoxtrafeet`.

```

1683 \newcommand*{\l@ddodoreinxtrafeet}{%
1684   \doreinxtrafeeti
1685   \doreinxtrafeetii}
1686

```

`\doreinextrafeetii` `\doreinextrafeetii` is the code for catering for the class 2 extra critical footnotes within `\@reinserts`. The implementation may well have to change.

```
1687 \newcommand*{\doreinextrafeetii}{%
1688   \ifvoid\Afootins\else\insert\Afootins{\unvbox\Afootins}\fi
1689   \ifvoid\Bfootins\else\insert\Bfootins{\unvbox\Bfootins}\fi
1690   \ifvoid\Cfootins\else\insert\Cfootins{\unvbox\Cfootins}\fi
1691   \ifvoid\Dfootins\else\insert\Dfootins{\unvbox\Dfootins}\fi
1692   \ifvoid\Efootins\else\insert\Efootins{\unvbox\Efootins}\fi
1693 }
1694
```

`\l@d@reinserts` And here is the modified version of `\@reinserts`.

```
1695 \gdef \l@d@reinserts{%
1696   \ifvoid\footins\else\insert\footins{\unvbox\footins}\fi
1697   \l@dd@doreinextrafeet
1698   \ifvbox\@kludgeins\insert\@kludgeins{\unvbox\@kludgeins}\fi
1699 }
1700
```

The memoir class does not use the ‘standard’ versions of `\@makecol` and `\@reinserts`, due to its sidebar insert. We had better add that code if memoir is used. (It can be awkward dealing with `\if` code within `\if` code, so don’t use `\ifl@d@memoir` here.)

```
1701 \@ifclassloaded{memoir}{%
    memoir is loaded so we use memoir’s built in hooks.
1702   \g@addto@macro{\m@mdoextrafeet}{\l@dd@extrafeet}%
1703   \g@addto@macro{\m@mdodoreinextrafeet}{\l@dd@doreinextrafeet}%
1704 }{%
    memoir has not been loaded, so redefine @makecol and @reinserts.
1705   \gdef\@makecol{\l@d@makecol}%
1706   \gdef\@reinserts{\l@d@reinserts}%
1707 }
1708
```

`\addfootins` Let’s make it easier for an author to create a new series by providing this macro, `\addfootins{<letter>}`, to add the series to the several lists.

```
1709 \newcommand*{\addfootins}[1]{%
1710   \footnormal{#1}
    Add it to the output.
1711   \g@addto@macro{\@opxtrafeetii}{%
1712     \ifvoid\@nameuse{#1footins}\else
1713       \@nameuse{#1footstart{#1}}\@nameuse{#1footgroup}{#1}\fi}
    Add it to the reinsertions.
1714   \g@addto@macro{\doreinextrafeetii}{%
1715     \ifvoid\@nameuse{#1footins}\else
1716       \insert\@nameuse{#1footins}{\unvbox\@nameuse{#1footins}}\fi}
```

Add it to minipages.

```
1717 \g@addto@macro{\l@dedbeginmini}{%
1718 \expandafter\let\csname #1footnote\endcsname = \@nameuse{mp#1footnote}}
```

And at the end of a minipage.

```
1719 \g@addto@macro{\l@dedendmini}{%
1720 \ifvoid\@nameuse{mp#1footins}\else\@nameuse{mpfootgroup#1{#1}}\fi}
1721 }
1722
```

It turns out that \@doclearpage also needs modifying.

\if@led@nofoot We have to check if there are any leftover feet. \@led@extranofeet is a hook for
\@led@extranofeet handling further footnotes.

```
1723 \newif\if@led@nofoot
1724 \newcommand*{\@led@extranofeet}{}
1725
```

```
1726 \@ifclassloaded{memoir}{%
```

If the memoir class is loaded we hook into its modified \@doclearpage.

\@mem@extranofeet

```
1727 \g@addto@macro{\@mem@extranofeet}{%
1728 \ifvoid\Afootins\else\@mem@nofootfalse\fi
1729 \ifvoid\Bfootins\else\@mem@nofootfalse\fi
1730 \ifvoid\Cfootins\else\@mem@nofootfalse\fi
1731 \ifvoid\Dfootins\else\@mem@nofootfalse\fi
1732 \ifvoid\Efootins\else\@mem@nofootfalse\fi
1733 \ifvoid\footinsA\else\@mem@nofootfalse\fi
1734 \ifvoid\footinsB\else\@mem@nofootfalse\fi
1735 \ifvoid\footinsC\else\@mem@nofootfalse\fi
1736 \@led@extranofeet}
1737 }{%
```

As memoir is not loaded we have to do it all here.

\@led@testifnofoot

```
\@doclearpage 1738 \newcommand*{\@led@testifnofoot}{%
1739 \@led@nofoottrue
1740 \ifvoid\footins\else\@led@nofootfalse\fi
1741 \ifvoid\Afootins\else\@led@nofootfalse\fi
1742 \ifvoid\Bfootins\else\@led@nofootfalse\fi
1743 \ifvoid\Cfootins\else\@led@nofootfalse\fi
1744 \ifvoid\Dfootins\else\@led@nofootfalse\fi
1745 \ifvoid\Efootins\else\@led@nofootfalse\fi
1746 \ifvoid\footinsA\else\@led@nofootfalse\fi
1747 \ifvoid\footinsB\else\@led@nofootfalse\fi
1748 \ifvoid\footinsC\else\@led@nofootfalse\fi
1749 \@led@extranofeet}
1750
```

```

1751 \renewcommand{\@docclearpage}{%
1752   \@led@testifnofoot
1753   \if@led@nofoot
1754     \setbox\@tempboxa\vsplit\@cclv to\z@ \unvbox\@tempboxa
1755     \setbox\@tempboxa\box\@cclv
1756     \xdef\@deferlist{\@toplist\@botlist\@deferlist}%
1757     \global \let \@toplist \@empty
1758     \global \let \@botlist \@empty
1759     \global \@colroom \@colht
1760     \ifx \@currlist\@empty
1761       \else
1762         \@latexerr{Float(s) lost}\@ehb
1763         \global \let \@currlist \@empty
1764       \fi
1765       \@makefcolumn\@deferlist
1766       \@whilesw\if@fcolmade \fi{\@opcol\@makefcolumn\@deferlist}%
1767       \if@twocolumn
1768         \if@firstcolumn
1769           \xdef\@dbldeferlist{\@dbltoplist\@dbldeferlist}%
1770           \global \let \@dbltoplist \@empty
1771           \global \@colht \textheight
1772           \begingroup
1773             \@dblfloatplacement
1774             \@makefcolumn\@dbldeferlist
1775             \@whilesw\if@fcolmade \fi{\@outputpage
1776               \@makefcolumn\@dbldeferlist}%
1777           \endgroup
1778         \else
1779           \vbox{}\clearpage
1780         \fi
1781       \fi
1782     \else
1783       \setbox\@cclv\vbox{\box\@cclv\vfil}%
1784       \l@makecol\@opcol
1785       \clearpage
1786     \fi}
1787 }
1788

```

24 Cross referencing

I have rewritten portions of the code in this section so that the LaTeX .aux file is used. This will also handle \included files.

Further, I have renamed some of the original EDMAC macros so that they do not clash with the LaTeX label/ref commands (EDMAC and LaTeX use very different mechanisms). In particular, the original EDMAC \label and \pageref have been renamed as \edlabel and \edpageref respectively.

You can mark a place in the text using a command of the form \edlabel{foo},

and later refer to it using the label `foo` by saying `\edpageref{foo}`, or `\lineref{foo}` or `\sublineref{foo}`. These reference commands will produce, respectively, the page, line and sub-line on which the `\edlabel{foo}` command occurred.

The reference macros warn you if a reference is made to an undefined label. If `foo` has been used as a label before, the `\edlabel{foo}` command will issue a complaint; subsequent `\edpageref` and `\lineref` commands will refer to the latest occurrence of `\label{foo}`.

`\labelref@list` Set up a new list, `\labelref@list`, to hold the page, line and sub-line numbers for each label.

```
1789 \list@create{\labelref@list}
```

`\zz@@@` A convenience macro to zero two labeling counters in one go.

```
1790 %% \newcommand*\zz@@@{000|000|000} % set three counters to zero in one go
1791 \newcommand*\zz@@@{000|000} % set two counters to zero in one go
1792
```

`\edlabel` The `\edlabel` command first writes a `\@lab` macro to the `\linenum@out` file. It then checks to see that the `\labelref@list` actually has something in it (if not, it creates a dummy entry), and pops the next value for the current label, storing it in `\label@refs`. Finally it defines the label to be `\empty` so that any future check will turn up the fact that it has been used.²⁸

This version of the original EDMAC `\label` uses `\@bsphack` and `\@esphack` to eliminate extra space problems and also the LaTeX write methods for the `.aux` file.

Jesse Billett²⁹ found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```
1793 \newcommand*\edlabel{[1]{\@bsphack
1794   \write\linenum@out{\string\@lab}%
1795   \ifx\labelref@list\empty
1796     \xdef\label@refs{\zz@@@}%
1797   \else
1798     \gl@p\labelref@list\to\label@refs
1799   \ifvmode
1800     \advance\label@refs
1801   \fi
1802   \fi
1803 %   \edef\next{\write\@aux{\string\l@dmake@labels\label@refs|{#1}}}%
1804 %   \next}
```

Use code from the kernel `\label` command to write the correct page number (it seems possible that the original EDMAC's `\page@num` scheme might also have had problems in this area).

```
1805 \protected@write\@auxout{}%
```

²⁸The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

²⁹(jdb43@cam.ac.uk) via the ctt thread 'ledmac cross referencing', 25 August 2003.

```

1806     {\string\l@dmake@labels\space\thepage|\label@refs|{#1}}%
1807     \@esphack}
1808

```

```

\advancelabel@refs
\labelrefsparseline 1809 %In cases where \cs{edlabel} is the first element in a paragraph, we have a problem with line counts,
\labelrefsparsesubline 1810 %Hence, we need to test \cs{edlabel} if it occurs at the start of a paragraph. To do so, we use \cs{
1811 %We do so using \cs[advancelabel@refs] command.
1812 \newcommand{\advancelabel@refs}{%
1813 \newcount\line%
1814 \ledmac@warning{\label@refs}
1815 \line=\expandafter\labelrefsparseline\label@refs%
1816 \advance\line by 1%
1817 \ifsublines@%
1818 \newcount\subline%
1819 \subline=\labelrefsparsesubline\label@refs%
1820 \advance\subline by 1%
1821 \def\label@refs{\the\line|\the\subline}%
1822 \else%
1823 \def\label@refs{\the\line|0}%
1824 \fi%
1825 }
1826 \def\labelrefsparseline#1|#2{#1}
1827 \def\labelrefsparsesubline#1|#2{#2}

```

`\l@dmake@labels` The `\l@dmake@labels` macro gets executed when the labels file is read. For each label it defines a macro, whose name is made up partly from the label you supplied, that contains the page, line and sub-line numbers. But first it checks to see whether the label has already been used (and complains if it has).

The initial use of `\newcommand` is to catch if `\l@dmake@labels` has been previously defined (by a class or package).

```

1828 \newcommand*{\l@dmake@labels}{%
1829 \def\l@dmake@labels#1|#2|#3|#4{%
1830   \expandafter\ifx\csname the@label#4\endcsname \relax\else
1831     \led@warn@DuplicateLabel{#4}%
1832   \fi
1833   \expandafter\gdef\csname the@label#4\endcsname{#1|#2|#3}%
1834   \ignorespaces}
1835

```

LaTeX reads the aux file at both the beginning and end of the document, so we have to switch off duplicate label checking after the first time the file is read.

```

1836 \AtBeginDocument{%
1837   \def\l@dmake@labels#1|#2|#3|#4{%
1838 }
1839

```

`\@lab` The `\@lab` command, which appears in the `\linenum@out` file, appends the current values of page, line and sub-line to the `\labelref@list`. These values are defined

by the earlier `\@page`, `\@l`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

LaTeX uses the page counter for page numbers. However, it appears that this is not the right place to grab the page number. That task is now done in the `\edlabel` macro. This version of `\@lab` appends just the current line and sub-line numbers to `\labelref@list`.

```
1840 \newcommand*{\@lab}{\xright@appenditem
1841   {\linenumrep{\line@num}|%
1842     \ifsublines@ \sublinenumrep{\subline@num}\else 0\fi}\to\labelref@list}
1843
```

`\edpageref` If the specified label exists, `\edpageref` gives its page number. For this reference command, as for the other two, a special version with prefix `x` is provided for use in places where the command is to be scanned as a number, as in `\linenum`. These special versions have two limitations: they don't print error messages if the reference is unknown, and they can't appear as the first label or reference command in the file; you must ensure that a `\edlabel` or a normal reference command appears first, or these `x`-commands will always return zeros. LaTeX already defines a `\pageref`, so changing the name to `\edpageref`.

```
1844 \newcommand*{\edpageref}[1]{\l@dref@undefined{#1}\l@dgetref@num{1}{#1}}
1845 \newcommand*{\xpageref}[1]{\l@dgetref@num{1}{#1}}
1846
```

`\lineref` If the specified label exists, `\lineref` gives its line number.

```
\xlineref 1847 \newcommand*{\lineref}[1]{\l@dref@undefined{#1}\l@dgetref@num{2}{#1}}
1848 \newcommand*{\xlineref}[1]{\l@dgetref@num{2}{#1}}
1849
```

`\sublineref` If the specified label exists, `\sublineref` gives its sub-line number.

```
\xsublineref 1850 \newcommand*{\sublineref}[1]{\l@dref@undefined{#1}\l@dgetref@num{3}{#1}}
1851 \newcommand*{\xsublineref}[1]{\l@dgetref@num{3}{#1}}
1852
```

The next three macros are used by the referencing commands above, and do the job of extracting the right numbers from the label macro that contains the page, line, and sub-line number.

`\l@dref@undefined` The `\l@dref@undefined` macro is called when you refer to a label with the normal referencing macros. Its argument is a label, and it just checks that the label has been defined.

```
1853 \newcommand*{\l@dref@undefined}[1]{%
1854   \expandafter\ifx\csname the@label#1\endcsname\relax
1855     \led@warn@RefUndefined{#1}%
1856   \fi}
1857
```

`\l@dgetref@num` Next, `\l@dgetref@num` fetches the number we want. It has two arguments: the first is simply a digit, specifying whether to fetch a page (1), line (2) or sub-line

(3) number. (This switching is done by calling `\l@ddlabel@parse`.) The second argument is the label-macro, which because of the `\@lab` macro above is defined to be a string of the type 123|456|789.

```

1858 \newcommand*{\l@dgetref@num}[2]{%
1859   \expandafter
1860   \ifx\csname the@label#2\endcsname \relax
1861     000%
1862   \else
1863     \expandafter\expandafter\expandafter
1864     \l@ddlabel@parse\csname the@label#2\endcsname| #1%
1865   \fi}
1866

```

`\l@ddlabel@parse` Notice that we slipped another | delimiter into the penultimate line of `\l@dgetref@num`, to keep the ‘switch-number’ separate from the reference numbers. This | is used as another parameter delimiter by `\l@ddlabel@parse`, which extracts the appropriate number from its first arguments. The |-delimited arguments consist of the expanded label-macro (three reference numbers), followed by the switch-number (1, 2, or 3) which defines which of the earlier three numbers to pick out. (It was earlier given as the first argument of `\l@dgetref@num`.)

```

1867 \newcommand*{\l@ddlabel@parse}{%
1868   \def\l@ddlabel@parse#1|#2|#3|#4{%
1869     \ifcase #4\relax
1870     \or #1%
1871     \or #2%
1872     \or #3%
1873   \fi}
1874

```

`\xxref` The `\xxref` command takes two arguments, both of which are labels, e.g., `\xxref{mouse}{elephant}`. It first does some checking to make sure that the labels do exist (if one doesn’t, those numbers are set to zero). Then it calls `\linenum` and sets the beginning page, line, and sub-line numbers to those of the place where `\label{mouse}` was placed, and the ending numbers to those at `\label{elephant}`. The point of this is to be able to manufacture footnote line references to passages which can’t be specified in the normal way as the first argument to `\critext` for one reason or another. Using `\xxref` in the second argument of `\critext` lets you set things up at least semi-automatically.

```

1875 \newcommand*{\xxref}[2]{%
1876   {\expandafter\ifx\csname the@label#1\endcsname
1877     \relax \expandafter\let\csname the@label#1\endcsname\zz@@@ \fi
1878   \expandafter\ifx\csname the@label#2\endcsname \relax
1879     \expandafter\let\csname the@label#2\endcsname\zz@@@ \fi
1880   \linenum{\csname the@label#1\endcsname|}%
1881   \csname the@label#2\endcsname}}
1882

```

`\edmakelabel` Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired; you can use the `\edmakelabel` macro make your own label.

For example, if you say ‘`\edmakelabel{elephant}{10|25|0}`’ you will have created a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. `\edmakelabel` takes a label, followed by a page and a line number(s) as arguments. LaTeX defines a `\makelabel` macro which is used in lists. I’ve changed the name to `\edmakelabel`.

```
1883 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\endcsname{#2}}
1884
```

(If you are only going to refer to such a label using `\xxref`, then you can omit entries in the same way as with `\linenum` (see pp. 75 and 55), since `\xxref` makes a call to `\linenum` in order to do its work.)

25 Endnotes

`\l@d@end` Endnotes of all varieties are saved up in a file, typically named `<jobname>.end`.
`\ifl@dend@` `\l@d@end` is the output stream number for this file, and `\ifl@dend@` is a flag that’s
`\l@dend@true` true when the file is open.

```
\l@dend@false 1885 \newwrite\l@d@end
1886 \newif\ifl@dend@
```

`\l@dend@open` `\l@dend@open` and `\l@dend@close` are the macros that are used to open and close
`\l@dend@close` the endnote file. Note that all our writing to this file is `\immediate`: all page and
line numbers for the endnotes are generated by the same mechanism we use for
the footnotes, so that there’s no need to defer any writing to catch information
from the output routine.

```
1887 \newcommand{\l@dend@open}[1]{\global\l@dend@true\immediate\openout\l@d@end=#1\relax}
1888 \newcommand{\l@dend@close}{\global\l@dend@false\immediate\closeout\l@d@end}
1889
```

`\l@dend@stuff` `\l@dend@stuff` is used by `\beginnumbering` to do everything that’s necessary for
the endnotes at the start of each section: it opens the `\l@d@end` file, if necessary,
and writes the section number to the endnote file.

```
1890 \newcommand{\l@dend@stuff}{%
1891 \ifl@dend@\relax\else
1892 \l@dend@open{<jobname>.end}%
1893 \fi
1894 \immediate\write\l@d@end{\string\l@d@section{<the>\section@num}}%
1895
```

`\Aendnote` The following five macros each function to write one endnote to the `.end` file.
`\Bendnote` Like the footnotes, these endnotes come in five series, A through E. We change
`\Cendnote` `\newlinechar` so that in the file every space becomes the start of a new line; this
`\Dendnote` generally ensures that a long note doesn’t exceed restrictions on the length of lines
`\Eendnote` in files.

```
1896 \newcommand*{\Aendnote}[1]{\newlinechar='40
1897 \immediate\write\l@d@end{\string\Aend%
```

```

1898         {\ifnumberedpar@l@d@nums\fi}%
1899         {\ifnumberedpar@tag\fi}{#1}}\ignorespaces}

1900 \newcommand*{\Bendnote}[1]{\newlinechar='40
1901     \immediate\write\l@d@end{\string\Bend%
1902         {\ifnumberedpar@l@d@nums\fi}%
1903         {\ifnumberedpar@tag\fi}{#1}}\ignorespaces}

1904 \newcommand*{\Cendnote}[1]{\newlinechar='40
1905     \immediate\write\l@d@end{\string\Cend%
1906         {\ifnumberedpar@l@d@nums\fi}%
1907         {\ifnumberedpar@tag\fi}{#1}}\ignorespaces}

1908 \newcommand*{\Dendnote}[1]{\newlinechar='40
1909     \immediate\write\l@d@end{\string\Dend%
1910         {\ifnumberedpar@l@d@nums\fi}%
1911         {\ifnumberedpar@tag\fi}{#1}}\ignorespaces}

1912 \newcommand*{\Eendnote}[1]{\newlinechar='40
1913     \immediate\write\l@d@end{\string\Eend%
1914         {\ifnumberedpar@l@d@nums\fi}%
1915         {\ifnumberedpar@tag\fi}{#1}}\ignorespaces}
1916

```

\Aend **\Aendnote** and the like write commands called **\Aend** and so on to the endnote file; these are analogous to the various **footfmt** commands above, and they take the same arguments. When we process this file, we'll want to pick out the notes of one series and ignore all the rest. To do that, we equate the **end** command for the series we want to **\endprint**, and leave the rest equated to **\@gobblethree**, which just skips over its three arguments.³⁰ The **\endprint** here is nearly identical in its functioning to **\normalfootfmt**.

\l@d@section The endnote file also contains **\l@d@section** commands, which supply the section numbers from the main text; standard **ledmac** does nothing with this information, but it's there if you want to write custom macros to do something with it.

```

1917 \def\endprint#1#2#3{{\notefontsetup{\notenumfont\printendlines#1}}%
1918     \enspace{\select@lemmafонт#1|#2}\enskip#3\par}}
1919 \providecommand*{\@gobblethree}[3]{ }
1920 \let\Aend=\@gobblethree
1921 \let\Bend=\@gobblethree
1922 \let\Cend=\@gobblethree
1923 \let\Dend=\@gobblethree
1924 \let\Eend=\@gobblethree
1925 \let\l@d@section=\@gobble
1926

```

\setprintendlines The **\printendlines** macro is similar to **\printlines** but is for printing endnotes rather than footnotes.

³⁰Christophe Hebeisen (christophe.hebeisen@a3.epfl.ch) emailed on 2003/11/05 to say he had found that **\@gobblethree** was also defined in the **amsfonts** package.

The principal difference between foot- and endnotes is that footnotes are printed on the page where they are specified but endnotes are printed at a different point in the document. We need an indication of the source of an endnote; `\setprintendlines` provides this by always printing the page number. The coding is slightly simpler than `\setprintlines`.

First of all, we print the second page number only if the ending page number is different from the starting page number.

```
1927 \newcommand*{\setprintendlines}[6]{%
1928   \l@dpnumfalse \l@ddashfalse
1929   \ifnum#4=#1 \else
1930     \l@dpnumtrue
1931     \l@ddashtrue
1932   \fi
```

We print the ending line number if: (1) we're printing the ending page number, or (2) it's different from the starting line number.

```
1933   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
1934   \ifnum#2=#5 \else
1935     \l@d@elintrue
1936     \l@ddashtrue
1937   \fi
```

We print the starting sub-line if it's nonzero.

```
1938   \l@d@ssubfalse
1939   \ifnum#3=0 \else
1940     \l@d@ssubtrue
1941   \fi
```

We print the ending sub-line if it's nonzero and: (1) it's different from the starting sub-line number, or (2) the ending line number is being printed.

```
1942   \l@d@eslfalse
1943   \ifnum#6=0 \else
1944     \ifnum#6=#3
1945       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
1946     \else
1947       \l@d@esltrue
1948       \l@ddashtrue
1949     \fi
1950   \fi}
```

`\printendlines` Now we're ready to print it all.

```
1951 \def\printendlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
1952   \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
```

The only subtlety left here is when to print a period between numbers. But the only instance in which this is tricky is for the ending sub-line number: it could be coming after the starting sub-line number (in which case we want only the dash) or after an ending line number (in which case we need to insert a period).

```
1953   \printpnum{#1} \linenumrep{#2}%
1954   \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
```

```

1955 \ifl@d@dash \endashchar\fi
1956 \ifl@d@pnum \printnpnum{#4}\fi
1957 \ifl@d@elin \linenumrep{#5}\fi
1958 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi
1959 \endgroup}
1960

```

`\printnpnum` A macro to print a page number in an endnote.

```

1961 \newcommand*{\printnpnum}[1]{p.#1) }
1962

```

`\doendnotes` `\doendnotes` is the command you use to print one series of endnotes; it takes one argument, the series letter of the note series you want to print.

```

1963 \newcommand*{\doendnotes}[1]{\l@dend@close
1964   \begingroup
1965     \makeatletter
1966     \expandafter\let\csname #1end\endcsname=\endprint
1967     \input\jobname.end
1968   \endgroup}

```

`\noendnotes` You can say `\noendnotes` before the first `\beginnumbering` in your file if you aren't going to be using any of the endnote commands: this will suppress the creation of an `.end` file. If you do have some lingering endnote commands in your file, the notes will be written to your terminal and to the log file.

```

1969 \newcommand*{\noendnotes}{\global\let\l@dend@stuff=\relax
1970   \global\chardef\l@d@end=16 }

```

26 Side notes

Regular `\marginpars` do not work inside numbered text — they don't produce any note but do put an extra unnumbered blank line into the text.

`\l@dold@xympar` Changing `\@xympar` a little at least ensures that `\marginpars` in numbered text do not disturb the flow.

```

1971 \let\l@dold@xympar\@xympar
1972 \renewcommand{\@xympar}{%
1973   \ifnumberedpar@
1974     \led@warn@NoMarginpars
1975     \@esphack
1976   \else
1977     \l@dold@xympar
1978   \fi}
1979

```

We provide side notes as replacement for `\marginpar` in numbered text.

`\sidenote@margin` These are the sidenote equivalents to `\line@margin` and `\linenummargin` for specifying which margin. The default is the right margin (opposite to the default for line numbers).

```

1980 \newcount\sidenote@margin
1981 \newcommand*{\sidenotemargin}[1]{%
1982   \l@getsidenote@margin{#1}%
1983   \ifnum\@l@tempcntb>\m@ne
1984     \global\sidenote@margin=\@l@tempcntb
1985   \fi}}
1986 \newcommand*{\l@getsidenote@margin}[1]{%
1987   \def\@tempa{#1}\def\@tempb{left}%
1988   \ifx\@tempa\@tempb
1989     \@l@tempcntb \z@
1990   \else
1991     \def\@tempb{right}%
1992     \ifx\@tempa\@tempb
1993       \@l@tempcntb \@ne
1994     \else
1995       \def\@tempb{outer}%
1996       \ifx\@tempa\@tempb
1997         \@l@tempcntb \tw@
1998       \else
1999         \def\@tempb{inner}%
2000         \ifx\@tempa\@tempb
2001           \@l@tempcntb \thr@@
2002         \else
2003           \led@warn@BadSidenotemargin
2004           \@l@tempcntb \m@ne
2005         \fi
2006       \fi
2007     \fi
2008   \fi}
2009 \sidenotemargin{right}
2010
```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox 2011 \newbox\l@dlp@rbox
2012 \newbox\l@drp@rbox
2013
```

`\ledlsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`, their distance from the text (initialised to `\linenumsep`, and the fonts used.

```

\ledlsnotesep 2014 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep 2015 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup 2016 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup 2017 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
2018 \newcommand*{\ledlsnotefontsetup}{\raggedleft\footnotesize}
2019 \newcommand*{\ledrsnotefontsetup}{\raggedright\footnotesize}
2020
```

`\ledleftnote` `\ledleftnote{<text>}` and `\ledrightnote{<text>}` are the user commands for left and right sidenotes. `\ledsidenote{<text>}` is the command for a moveable sidenote.

```
\ledsidenote 2021 \newcommand*{\ledleftnote}[1]{\edtext{}{\l@dlsnote{#1}}}
2022 \newcommand*{\ledrightnote}[1]{\edtext{}{\l@drsnote{#1}}}
2023 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcsnote{#1}}}
2024
2025
```

`\l@dlsnote` The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminis-

`\l@drsnote` cent of the critical footnotes code.

```
\l@dcsnote 2026 \newif\ifrightrightnoteup
2027 \rightnoteuptrue
2028 \newcommand*{\l@dlsnote}[1]{%
2029 \ifnumberedpar@
2030 \xright@appenditem{\noexpand\l@dlsnote{#1}}{%
2031 \to\inserts@list
2032 \global\advance\insert@count \@ne
2033 \fi\ignorespaces}
2034 \newcommand*{\l@drsnote}[1]{%
2035 \ifnumberedpar@
2036 \xright@appenditem{\noexpand\l@drsnote{#1}}{%
2037 \to\inserts@list
2038 \global\advance\insert@count \@ne
2039 \fi\ignorespaces}
2040 \newcommand*{\l@dcsnote}[1]{%
2041 \ifnumberedpar@
2042 \xright@appenditem{\noexpand\l@dcsnote{#1}}{%
2043 \to\inserts@list
2044 \global\advance\insert@count \@ne
2045 \fi\ignorespaces}
2046
```

`\vl@dlsnote` Put the left/right text into boxes, but just save the moveable text.

```
\vl@drsnote 2047 \newcommand*{\vl@dlsnote}[1]{\setl@dlp@rbox{#1}}
\vl@dcsnote 2048 \newcommand*{\vl@drsnote}[1]{\setl@drp@rbox{#1}}
2049 \newcommand*{\vl@dcsnote}[1]{\gdef\l@dcsnotetext{#1}}
2050
```

`\setl@dlp@rbox` `\setl@dlp@rbox{<lednums>}{<tag>}{<text>}` puts `<text>` into the `\l@dlp@rbox` box.

`\setl@drp@rbox` And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```
2051 \newcommand*{\setl@dlp@rbox}[1]{%
2052 {\parindent\z@\hspace=\ledlsnotewidth\ledlsnotefontsetup
2053 \global\setbox\l@dlp@rbox
2054 \ifleftnoteup
2055 =\vbox to\z@{\vss #1}%
2056 \else
2057 =\vbox to 0.70\baselineskip{\strut#1\vss}%
2058 \fi}}
```

```

2059 %% \global\setbox\l@dlp@rbox=\vbox to\z@{#3\vss}}}% aligns on top line
2060 \newcommand*{\setl@drp@rbox}[1]{%
2061   {\parindent\z@\hspace=\ledrsnotewidth\ledrsnotefontsetup
2062     \global\setbox\l@drp@rbox
2063     \ifrightnoteup
2064       =\vbox to\z@{\vss#1}%
2065     \else
2066       =\vbox to0.7\baselineskip{\strut#1\vss}%
2067     \fi}}
2068 \newif\ifleftnoteup
2069 \leftnoteuptrue

\savel@dcstnote Save the moveable note text in \l@dcstnotetext.
\l@dcstnotetext 2070 \newcommand*{\savel@dcstnote}[3]{%
2071   \gdef\l@dcstnotetext{#3}}
2072

\affixside@note This macro puts any moveable sidenote text into the left or right sidenote box, de-
                  pending on which margin it is meant to go in. It's a very much stripped down version
                  of \affixlin@num.
2073 \newcommand*{\affixside@note}{%
2074   \gdef\@templ@d{}%
2075   \ifx\@templ@d\l@dcstnotetext \else
2076     \if@twocolumn
2077       \if@firstcolumn
2078         \setl@dlp@rbox{\l@dcstnotetext}%
2079       \else
2080         \setl@drp@rbox{\l@dcstnotetext}%
2081       \fi
2082     \else
2083       \@l@dttempcntb=\sidenote@margin
2084       \ifnum\@l@dttempcntb>\@ne
2085         \advance\@l@dttempcntb by\page@num
2086       \fi
2087       \ifodd\@l@dttempcntb
2088         \setl@drp@rbox{\l@dcstnotetext}%
2089       \else
2090         \setl@dlp@rbox{\l@dcstnotetext}%
2091       \fi
2092     \fi
2093   \fi}
2094

```

27 Familiar footnotes

The original EDMAC provided the five series of critical footnotes, and LaTeX provides a single numbered footnote. The ledmac package uses the EDMAC mechanism to provide a few series of numbered footnotes.

First, though, the footmisc package has an option whereby two or more consecutive `\footnotes` have their marks separated by commas. This seems such a useful ability that it is provided automatically by ledmac.

`\multiplefootnotemarker` These macros may have been defined by the memoir class, are provided by the footmisc package and perhaps by other footnote packages.

```
2095 \providecommand*\multiplefootnotemarker}{3sp}
2096 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
2097
```

`\m@mmf@prepare` A pair of self-cancelling kerns. This may have been defined in the memoir class.

```
2098 \providecommand*\m@mmf@prepare}{%
2099   \kern-\multiplefootnotemarker
2100   \kern\multiplefootnotemarker\relax}
```

`\m@mmf@check` This may have been defined in the memoir class. If it recognises the last kern as `\multiplefootnotemarker` it typesets `\multfootsep`.

```
2101 \providecommand*\m@mmf@check}{%
2102   \ifdim\lastkern=\multiplefootnotemarker\relax
2103     \edef\x@sf{\the\spacefactor}%
2104     \unkern
2105     \multfootsep
2106     \spacefactor\x@sf\relax
2107   \fi}
2108
```

We have to modify `\@footnotetext` and `\@footnotemark`. However, if memoir is used the modifications have already been made.

```
2109 \ifclassloaded{memoir}{}{%
```

`\@footnotetext` Add `\m@mmf@prepare` at the end of `\@footnotetext`.

```
2110 \let\l@dold@footnotetext\@footnotetext
2111 \renewcommand{\@footnotetext}[1]{%
2112   \l@dold@footnotetext{#1}%
2113   \m@mmf@prepare}
```

`\@footnotemark` Modify `\@footnotemark` to cater for adjacent `\footnotes`.

```
2114 \renewcommand*\@footnotemark}{%
2115   \leavevmode
2116   \ifhmode
2117     \edef\x@sf{\the\spacefactor}%
2118     \m@mmf@check
2119     \nobreak
2120   \fi
2121   \@makefnmark
2122   \m@mmf@prepare
2123   \ifhmode\spacefactor\x@sf\fi
2124   \relax}
```

Finished the modifications for the non-memoir case.

```
2125 }
2126
```

`\l@doldold@footnotetext` In order to enable the regular `\footnotes` in numbered text we have to play around
`\@footnotetext` with its `\@footnotetext`, using different forms for when in numbered or regular text.

```
2127 \let\l@doldold@footnotetext\@footnotetext
2128 \renewcommand{\@footnotetext}[1]{%
2129   \ifnumberedpar@
2130     \edtext{}{\l@dbfnote{#1}}%
2131   \else
2132     \l@doldold@footnotetext{#1}%
2133   \fi}
```

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original
`\vl@dbfnote` `\@footnotetext`.

```
2134 \newcommand{\l@dbfnote}[1]{%
2135   \ifnumberedpar@
2136     \xright@appenditem{\noexpand\vl@dbfnote{#1}}{\@thefnmark}}%
2137     \to\inserts@list
2138   \global\advance\insert@count \@ne
2139   \fi\ignorespaces}
2140 \newcommand{\vl@dbfnote}[2]{%
2141   \def\@thefnmark{#2}%
2142   \l@doldold@footnotetext{#1}}
2143
```

Now we can get on with providing the extra series of numbered footnotes. The general naming convention is to add an uppercase letter, denoting the series, at the end of macro names (the EDMAC series have an uppercase letter at the start of macro names).

First we'll give all the code for the A series, then the much more limited code for defining additional series.

27.1 The A series footnotes

`\footnoteA` `\footnoteA{<text>}` is the user level command.

```
2144 \newcommand{\footnoteA}[1]{%
2145   \stepcounter{footnoteA}%
2146   \protected@xdef\@thefnmarkA{\thefootnoteA}%
2147   \@footnotemarkA
2148   \vfootnoteA{A}{#1}\m@mmf@prepare}
2149
```

`\footinsA` The insert for the A series.

```
2150 \newinsert\footinsA
```

```

\c@footnoteA The A series counter.
\thefootnoteA 2151 \newcounter{footnoteA}
                2152 \renewcommand{\thefootnoteA}{\arabic{footnoteA}}
                2153

\footfootmarkA This macro typesets the A series marker at the start of the footnote text (where it
                appears at the foot of the page).
                2154 \newcommand*{\footfootmarkA}{\textsuperscript{\thefootnoteA}}
                2155

\mpfootnoteA The extras for minipages.
\mpfootinsA 2156 \newcommand{\mpfootnoteA}[1]{%
                2157 \stepcounter{footnoteA}%
                2158 \protected@xdef\thefnmarkA{\thefootnoteA}%
                2159 \@footnotemarkA
                2160 \mpvfootnoteA{A}{#1}\m@mmf@prepare}
                2161 \newinsert\mpfootinsA
                2162

```

We have to specify the default footnote style for the A series. This is done later.

That completes the specific macros that have to be specified for the A series. Similar ones are required for any other series.

27.2 Footnote formats

Some of the code for the various formats is remarkably similar to that in section 22.3.

The following macros generally set things up for the 'standard' footnote format.

```

\prebodyfootmark Two convenience macros for use by \...@footnotemark... macros.
\postbodyfootmark 2163 \newcommand*{\prebodyfootmark}{%
                2164 \leavevmode
                2165 \ifhmode
                2166 \edef\x@sf{\the\spacefactor}%
                2167 \m@mmf@check
                2168 \nobreak
                2169 \fi}
                2170 \newcommand{\postbodyfootmark}{%
                2171 \m@mmf@prepare
                2172 \ifhmode\spacefactor\x@sf\fi\relax}
                2173

\normal@footnotemarkX \normal@footnotemarkX{<series>} sets up the typesetting of the marker at the point
                where the footnote is called for.
                2174 \newcommand*{\normal@footnotemarkX}[1]{%
                2175 \prebodyfootmark
                2176 \@nameuse{bodyfootmark#1}%
                2177 \postbodyfootmark}
                2178

```

`\normalbodyfootmarkX` The `\normalbodyfootmarkX{<series>}` really typesets the in-text marker. The style is the normal superscript.

```
2179 \newcommand*{\normalbodyfootmarkX}[1]{%
2180   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
```

`\normalvfootnoteX` `\normalvfootnoteX{<series>}{<text>}` does the `\insert` for the `<series>` and calls the series' `\footfmt...` to format the `<text>`.

```
2181 \newcommand*{\normalvfootnoteX}[2]{%
2182   \insert\@nameuse{footins#1}\bgroup
2183     \notefontsetup
2184     \footsplitskips
2185     \spaceskip=\z@skip \xspaceskip=\z@skip
2186     \@nameuse{footfmt#1}{#1}{#2}\egroup}
2187
```

`\mpnormalvfootnoteX` The minipage version.

```
2188 \newcommand*{\mpnormalvfootnoteX}[2]{%
2189   \global\setbox\@nameuse{mpfootins#1}\vbox{%
2190     \unvbox\@nameuse{mpfootins#1}
2191     \notefontsetup
2192     \hsize\columnwidth
2193     \@parboxrestore
2194     \color@begingroup
2195     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
2196
```

`\normalfootfmtX` `\normalfootfmtX{<series>}{<text>}` typesets the footnote text, prepended by the marker.

```
2197 \newcommand*{\normalfootfmtX}[2]{%
2198   \ledsetnormalparstuff
2199   {\notenumfont\@nameuse{footfootmark#1}\strut%\enspace
2200    #2\strut\par}}
2201
```

`\normalfootfootmarkX` `\normalfootfootmarkX{<series>}` is called by `\normalfootfmtX` to typeset the footnote marker in the footer before the footnote text.

```
2202 \newcommand*{\normalfootfootmarkX}[1]{%
2203   \textsuperscript{\@nameuse{@thefnmark#1}}}
2204
```

`\normalfootstartX` `\normalfootstartX{<series>}` is the `<series>` footnote starting macro used in the output routine.

```
2205 \newcommand*{\normalfootstartX}[1]{%
2206   \vskip\skip\@nameuse{footins#1}%
2207   \leftskip=\z@
2208   \rightskip=\z@
2209   \@nameuse{footnoterule#1}}
2210
```

`\normalfootnoteruleX` The rule drawn before the footnote series group.

```
2211 \let\normalfootnoteruleX=\footnoterule
2212
```

`\normalfootgroupX` `\normalfootgroupX{<series>}` sends the contents of the `<series>` insert box to the output page without alteration.

```
2213 \newcommand*{\normalfootgroupX}[1]{%
2214   \unvbox\@nameuse{footins#1}}
2215
```

`\mpnormalfootgroupX` The minipage version.

```
2216 \newcommand*{\mpnormalfootgroupX}[1]{%
2217   \vskip\skip\@nameuse{mpfootins#1}
2218   \normalcolor
2219   \@nameuse{footnoterule#1}
2220   \unvbox\@nameuse{mpfootins#1}}
2221
```

`\normalbfnoteX`

```
2222 \newcommand{\normalbfnoteX}[2]{%
2223   \ifnumberedpar@
2224     \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}}{\@nameuse{thefootnote#1}}}%
2225     \to\inserts@list
2226     \global\advance\insert@count \@ne
2227   \fi\ignorespaces}
2228
```

`\vbfnoteX`

```
2229 \newcommand{\vbfnoteX}[3]{%
2230   \@namedef{@thefnmark#1}{#3}%
2231   \@nameuse{regvfootnote#1}{#1}{#2}}
2232
```

`\vnumfootnoteX`

```
2233 \newcommand{\vnumfootnoteX}[2]{%
2234   \ifnumberedpar@
2235     \edtext{}{\normalbfnoteX{#1}{#2}}%
2236   \else
2237     \@nameuse{regvfootnote#1}{#1}{#2}%
2238   \fi}
2239
```

`\footnormalX` `\footnormalX{<series>}` initialises the settings for the `<series>` footnotes. This should always be called for each series.

```
2240 \newcommand*{\footnormalX}[1]{%
2241   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
2242   \@namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
2243   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}}
```

```

2244 \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
2245 \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
2246 \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
2247 \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
2248 \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
2249 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
2250 \count\csname footins#1\endcsname=1000
2251 \dimen\csname footins#1\endcsname=\ledfootinsdim
2252 \skip\csname footins#1\endcsname=1.2em \@plus .6em \@minus .6em

```

Additions for minipages.

```

2253 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
2254 \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
2255 \count\csname mpfootins#1\endcsname=1000
2256 % \dimen\csname mpfootins#1\endcsname=0.8\vsiz
2257 \dimen\csname mpfootins#1\endcsname=\ledfootinsdim
2258 \skip\csname mpfootins#1\endcsname=1.2em \@plus .6em \@minus .6em
2259 }
2260

```

27.2.1 Two column footnotes

The following macros set footnotes in two columns. It is assumed that the length of each footnote is less than the column width.

```

\foottwocolX \foottwocolX{<series>}
2261 \newcommand*{\foottwocolX}[1]{%
2262 \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
2263 \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
2264 \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
2265 \twocolfootsetupX{#1}
2266 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
2267 \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
2268 \mptwocolfootsetupX{#1}}
2269

\twocolfootsetupX \twocolfootsetupX{<series>}
\mptwocolfootsetupX 2270 \newcommand*{\twocolfootsetupX}[1]{%
2271 \count\csname footins#1\endcsname 500
2272 \multiply\dimen\csname footins#1\endcsname by \tw@}
2273 \newcommand*{\mptwocolfootsetupX}[1]{%
2274 \count\csname mpfootins#1\endcsname 500
2275 \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
2276

\twocolvfootnoteX \twocolvfootnoteX{<series>}
2277 \newcommand*{\twocolvfootnoteX}[2]{%
2278 \insert\csname footins#1\endcsname\bgroup
2279 \notefontsetup
2280 \footssplitsskip

```

```

2281 \spaceskip=\z@skip \xspaceskip=\z@skip
2282 \@nameuse{footfmt#1}{#1}{#2}\egroup
2283
\twocolfootfmtX \twocolfootfmtX{<series>}
2284 \newcommand*{\twocolfootfmtX}[2]{%
2285 \normal@pars
2286 \hsize .45\hsize
2287 \parindent=\z@
2288 %% \parfillskip=0pt \@plus 1fil
2289 \tolerance=5000\relax
2290 \raggedright
2291 \leavevmode
2292 {\notenumfont\@nameuse{footfootmark#1}\strut%\enspace
2293 #2\strut\par}\allowbreak}
2294
\twocolfootgroupX \twocolfootgroupX{<series>}
\mptwocolfootgroupX 2295 \newcommand*{\twocolfootgroupX}[1]{\notefontsetup
2296 \splittopskip=\ht\strutbox
2297 \expandafter
2298 \rigidbalance\csname footins#1\endcsname \tw@ \splittopskip}}
2299 \newcommand*{\mptwocolfootgroupX}[1]{%
2300 \vskip\skip\@nameuse{mpfootins#1}
2301 \normalcolor
2302 \@nameuse{footnoterule#1}
2303 \splittopskip=\ht\strutbox
2304 \expandafter
2305 \rigidbalance\csname mpfootins#1\endcsname \tw@ \splittopskip}}
2306

```

27.2.2 Three column footnotes

The following macros set footnotes in three columns. It is assumed that the length of each footnote is less than the column width.

```

\footthreecolX \footthreecolX{<series>}
2307 \newcommand*{\footthreecolX}[1]{%
2308 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
2309 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
2310 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
2311 \threecolfootsetupX{#1}
2312 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
2313 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
2314 \mpthreecolfootsetupX{#1}}
2315
\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX 2316 \newcommand*{\threecolfootsetupX}[1]{%

```

```

2317 \count\csname footins#1\endcsname 333
2318 \multiply\dimen\csname footins#1\endcsname by \thr@@
2319 \newcommand*\mpthreecolfootsetupX}[1]{%
2320 \count\csname mpfootins#1\endcsname 333
2321 \multiply\dimen\csname mpfootins#1\endcsname by \thr@@
2322
\threecolvfootnoteX \threecolvfootnoteX{<series>}{<text>}
2323 \newcommand*\threecolvfootnoteX}[2]{%
2324 \insert\csname footins#1\endcsname\bgroup
2325 \notefontsetup
2326 \footsplitskips
2327 \@nameuse{footfmt#1}{#1}{#2}\egroup}
2328
\threecolfootfmtX \threecolfootfmtX{<series>}
2329 \newcommand*\threecolfootfmtX}[2]{%
2330 \normal@pars
2331 \hsize .3\hsize
2332 \parindent=\z@
2333 %% \parfillskip=0pt \@plus 1fil
2334 \tolerance=5000\relax
2335 \raggedright
2336 \leavevmode
2337 {\notenumfont\@nameuse{footfootmark#1}\strut%\enspace
2338 #2\strut\par}\allowbreak}
2339
\threecolfootgroupX \threecolfootgroupX{<series>}
\mpthreecolfootgroupX 2340 \newcommand*\threecolfootgroupX}[1]{\notefontsetup
2341 \splittopskip=\ht\strutbox
2342 \expandafter
2343 \rigidbalance\csname footins#1\endcsname \thr@@ \splittopskip}}
2344 \newcommand*\mpthreecolfootgroupX}[1]{%
2345 \vskip\skip\@nameuse{mpfootins#1}
2346 \normalcolor
2347 \@nameuse{footnoterule#1}
2348 \splittopskip=\ht\strutbox
2349 \expandafter
2350 \rigidbalance\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
2351

```

27.2.3 Paragraphed footnotes

The following macros set footnotes as one paragraph.

```

\footparagraphX \footparagraphX{<series>}
2352 \newcommand*\footparagraphX}[1]{%
2353 \expandafter\let\csname footstart#1\endcsname=\parafootstartX

```



```

2354 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
2355 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
2356 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
2357 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
2358 \count\csname footins#1\endcsname=1000
2359 \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
2360 \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
2361 \count\csname mpfootins#1\endcsname=1000
2362 \para@footsetupX{#1}}
2363

```

`\para@footsetupX` `\para@footsetupX{<series>}`

```

2364 \newcommand*{\para@footsetupX}[1]{\notefontsetup
2365 \dimen0=\baselineskip
2366 \multiply\dimen0 by 1024
2367 \divide\dimen0 by \hsize \multiply\dimen0 by \footfudgefiddle\relax
2368 \expandafter
2369 \xdef\csname footfudgefactor#1\endcsname{%
2370 \expandafter\strip@pt\dimen0 }}
2371

```

`\parafootstartX` `\parafootstartX{<series>}`

```

2372 \newcommand*{\parafootstartX}[1]{%
2373 \vskip\skip\@nameuse{footins#1}%
2374 \leftskip=\z@
2375 \rightskip=\z@
2376 \parindent=\z@
2377 \vskip\skip\@nameuse{footins#1}%
2378 \@nameuse{footnoterule#1}}
2379

```

`\para@vfootnoteX` `\para@vfootnoteX{<series>}{<text>}`

```

\mppara@vfootnoteX 2380 \newcommand*{\para@vfootnoteX}[2]{%
2381 \insert\csname footins#1\endcsname
2382 \bgroup
2383 \notefontsetup
2384 \footsplitskips
2385 \setbox0=\vbox{\hsize=\maxdimen
2386 \noindent\@nameuse{footfmt#1}{#1}{#2}}%
2387 \setbox0=\hbox{\unvbox0}%
2388 \dp0=\z@
2389 \ht0=\csname footfudgefactor#1\endcsname\wd0
2390 \box0
2391 \penalty0
2392 \egroup}
2393 \newcommand*{\mppara@vfootnoteX}[2]{%
2394 \global\setbox\@nameuse{mpfootins#1}\vbox{%
2395 \unvbox\@nameuse{mpfootins#1}
2396 \notefontsetup

```

```

2397 \footsplitskips
2398 \setbox0=\vbox{\hsize=\maxdimen
2399 \noindent\color@begingroup\@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
2400 \setbox0=\hbox{\unvxh0}%
2401 \dp0=\z@
2402 \ht0=\csname footfudgefactor#1\endcsname\wd0
2403 \box0
2404 \penalty0}}
2405

\parafootfmtX \parafootfmtX{<series>}
2406 \newcommand*{\parafootfmtX}[2]{%
2407 \insertparafootftmsep
2408 \ledsetnormalparstuff
2409 {\notenumfont\@nameuse{footfootmark#1}\strut%\enspace
2410 #2\penalty-10}}
2411

\para@footgroupX \para@footgroupX{<series>}
\mppara@footgroupX 2412 \newcommand*{\para@footgroupX}[1]{%
2413 \unvbox\csname footins#1\endcsname
2414 \makehboxofhboxes
2415 \setbox0=\hbox{\unhbox0 \removehboxes}%
2416 \notefontsetup
2417 \noindent\unhbox0\par}
2418 \newcommand*{\mppara@footgroupX}[1]{%
2419 \vskip\skip\@nameuse{mpfootins#1}
2420 \normalcolor
2421 \@nameuse{footnoterule#1}
2422 \unvbox\csname mpfootins#1\endcsname
2423 \makehboxofhboxes
2424 \setbox0=\hbox{\unhbox0 \removehboxes}%
2425 \notefontsetup
2426 \noindent\unhbox0\par}}
2427

```

27.3 Other series footnotes

Other series, such as B, are provided here.

```

\footnoteB \footnoteB{<text>} is the user command for a series B footnote.
2428 \newcommand{\footnoteB}[1]{%
2429 \stepcounter{footnoteB}%
2430 \protected@xdef\@thefnmarkB{\thefootnoteB}%
2431 \@footnotemarkB
2432 \vfootnoteB{B}{#1}\m@mmf@prepare}
2433

\c@footnoteB
\thefootnoteB

```

```

2434 \newcounter{footnoteB}
2435 \renewcommand{\thefootnoteB}{\arabic{footnoteB}}
2436
\footinsB
2437 \newinsert\footinsB
2438
\mpfootnoteB The extras for minipages.
\mpfootinsB 2439 \newcommand{\mpfootnoteB}[1]{%
2440 \stepcounter{footnoteB}%
2441 \protected@xdef\@thefnmarkB{\thefootnoteB}%
2442 \@footnotemarkB
2443 \mpvfootnoteB{B}{#1}\m@mmf@prepare}
2444 \newinsert\mpfootinsB
2445

\footnoteC \footnoteC{\text} is the user command for a series C footnote.
2446 \newcommand{\footnoteC}[1]{%
2447 \stepcounter{footnoteC}%
2448 \protected@xdef\@thefnmarkC{\thefootnoteC}%
2449 \@footnotemarkC
2450 \vfootnoteC{C}{#1}\m@mmf@prepare}

\c@footnoteC
\thefootnoteC 2451 \newcounter{footnoteC}
\footinsC 2452 \renewcommand{\thefootnoteC}{\arabic{footnoteC}}
2453 \newinsert\footinsC
2454

\mpfootnoteC The extras for minipages.
\mpfootinsC 2455 \newcommand{\mpfootnoteC}[1]{%
2456 \stepcounter{footnoteC}%
2457 \protected@xdef\@thefnmarkC{\thefootnoteC}%
2458 \@footnotemarkC
2459 \mpvfootnoteC{C}{#1}\m@mmf@prepare}
2460 \newinsert\mpfootinsC
2461

Don't forget to initialise the series.
2462 \footnormalX{A}
2463 \footnormalX{B}
2464 \footnormalX{C}
2465

\doxtrafeeti We have to add all the new kinds of familiar footnotes to the output routine. These
\doreinxtrafeeti are the class 1 feet.
2466 \newcommand*\doxtrafeeti{%

```

```

2467 \setbox\@outputbox \vbox{%
2468   \unvbox\@outputbox
2469   \ifvoid\footinsA\else\footstartA{A}\footgroupA{A}\fi
2470   \ifvoid\footinsB\else\footstartB{B}\footgroupB{B}\fi
2471   \ifvoid\footinsC\else\footstartC{C}\footgroupC{C}\fi
2472 }}
2473
2474 \newcommand{\doreinextrafeeti}{%
2475   \ifvoid\footinsA\else\insert\footinsA{\unvbox\footinsA}\fi
2476   \ifvoid\footinsB\else\insert\footinsB{\unvbox\footinsB}\fi
2477   \ifvoid\footinsC\else\insert\footinsC{\unvbox\footinsC}\fi
2478 }
2479

```

`\addfootinsX` Make life just a little easier for those who want additional series of class 1 footnotes.

```

2480 \newcommand*{\addfootinsX}[1]{%
2481   \footnormalX{#1}%
2482   \g@addto@macro{\doextrafeeti}{%
2483     \setbox\@outputbox \vbox{%
2484       \unvbox\@outputbox
2485       \ifvoid\@nameuse{footins#1}\else
2486         \@nameuse{footstart#1}{#1}\@nameuse{footgroup#1}{#1}\fi}%
2487   \g@addto@macro{\doreinextrafeeti}{%
2488     \ifvoid\@nameuse{footins#1}\else
2489       \insert\@nameuse{footins#1}{\unvbox\@nameuse{footins#1}\fi}%
2490   \g@addto@macro{\l@dfambeginmini}{%
2491     \expandafter\expandafter\expandafter\let\expandafter\expandafter
2492       \csname footnote#1\endcsname \csname mpfootnote#1\endcsname}%
2493   \g@addto@macro{\l@dfamendmini}{%
2494     \ifvoid\@nameuse{mpfootins#1}\else\@nameuse{mpfootgroup#1}{#1}\fi}%
2495 }

```

28 Minipages and such

We can put footnotes into minipages. The preparatory code has been set up earlier, all that remains is to ensure that it is available inside a minipage box. This requires some alteration to the kernel code, specifically the `\@iiminipage` and `\endminipage` macros. We'll arrange this so that additional series can be easily added.

`\l@dfetbeginmini` These will be the hooks in `\@iiminipage` and `\endminipage` They can be extended
`\l@dfetendmini` to handle other things if necessary.

```

2496 \newcommand*{\l@dfetbeginmini}{\l@dedbeginmini\l@dfambeginmini}
2497 \newcommand*{\l@dfetendmini}{\l@dedendmini\l@dfamendmini}
2498

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.
`\l@dedendmini` They can be extended to cater for additional series.

```

2499 \newcommand*{\l@dedbeginmini}{%

```

```

2500 \let\Afootnote=\mpAfootnote%
2501 \let\Bfootnote=\mpBfootnote%
2502 \let\Cfootnote=\mpCfootnote%
2503 \let\Dfootnote=\mpDfootnote%
2504 \let\Efootnote=\mpEfootnote%
2505 \newcommand*{\l@dedendmini}{%
2506 \ifvoid\mpAfootins\else\mpAfootgroup{A}\fi%
2507 \ifvoid\mpBfootins\else\mpBfootgroup{B}\fi%
2508 \ifvoid\mpCfootins\else\mpCfootgroup{C}\fi%
2509 \ifvoid\mpDfootins\else\mpDfootgroup{D}\fi%
2510 \ifvoid\mpEfootins\else\mpEfootgroup{E}\fi}
2511

```

`\l@dfambeginmini` These handle the initiation and closure of familiar footnotes in a minipage environment.

`\l@dfamendmini` They can be extended to cater for additional series.

```

2512 \newcommand*{\l@dfambeginmini}{%
2513 \let\footnoteA=\mpfootnoteA%
2514 \let\footnoteB=\mpfootnoteB%
2515 \let\footnoteC=\mpfootnoteC%
2516 \newcommand*{\l@dfamendmini}{%
2517 \ifvoid\mpfootinsA\else\mpfootgroupA{A}\fi%
2518 \ifvoid\mpfootinsB\else\mpfootgroupB{B}\fi%
2519 \ifvoid\mpfootinsC\else\mpfootgroupC{C}\fi}
2520

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

2521 \def\@iiiminipage#1#2[#3]#4{%
2522 \leavevmode
2523 \@pboxswfalse
2524 \setlength\@tempdima{#4}%
2525 \def\@mpargs{#1}#2[#3]{#4}}%
2526 \setbox\@tempboxa\vbox\bgroup
2527 \color@begingroup
2528 \hsize\@tempdima
2529 \textwidth\hsize \columnwidth\hsize
2530 \@parboxrestore
2531 \def\@mpfn{\mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
2532 \let\@footnotetext\@mpfootnotetext

```

The next line is our addition to the original.

```

2533 \l@dfetbeginmini% added
2534 \let\@listdepth\@mplistdepth \@mplistdepth\z@
2535 \@minipagerestore
2536 \@setminipage}
2537

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

2538 \def\endminipage{%
2539 \par
2540 \unskip

```

```

2541 \ifvoid\@mpfootins\else
2542   \l@dunboxmpfoot
2543 \fi

```

The next line is our addition to the original.

```

2544 \l@dfeetendmini%           added
2545 \@minipagefalse
2546 \color@endgroup
2547 \egroup
2548 \expandafter\@iiiparbox\@mpargs{\unvbox\@tempboxa}}
2549

```

`\l@dunboxmpfoot`

```

2550 \newcommand*{\l@dunboxmpfoot}{%
2551   \vskip\skip\@mpfootins
2552   \normalcolor
2553   \footnoterule
2554   \unvbox\@mpfootins}
2555

```

`ledgroup` This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, fixed width minipage.

```

2556 \newenvironment{ledgroup}{%
2557   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
2558   \let\@footnotetext\@mpfootnotetext
2559   \l@dfeetbeginmini%
2560 }{%
2561   \par
2562   \unskip
2563   \ifvoid\@mpfootins\else
2564     \l@dunboxmpfoot
2565   \fi
2566   \l@dfeetendmini%
2567 }
2568

```

`ledgroupsize` `\begin{ledgroupsize}[\langle pos \rangle]{\langle width \rangle}`

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable *width* minipage. The optional *pos* controls the sideways position of numbered text.

```

2569 \newenvironment{ledgroupsize}[2][1]{%

```

Set the various text measures.

```

2570   \hsize #2\relax
2571   %% \textwidth #2\relax
2572   %% \columnwidth #2\relax

```

Initialize fills for centering.

```

2573   \let\ledllfill\hfil

```

```

2574 \let\ledrlfill\hfil
2575 \def\@tempa{#1}\def\@tempb{1}%
    Left adjusted numbered lines
2576 \ifx\@tempa\@tempb
2577 \let\ledllfill\relax
2578 \else
2579 \def\@tempb{r}%
2580 \ifx\@tempa\@tempb
    Right adjusted numbered lines
2581 \let\ledrlfill\relax
2582 \fi
2583 \fi
    Set up the footnoting.
2584 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
2585 \let\@footnotetext\@mpfootnotetext
2586 \l@dfetbeginmini%
2587 }{f%
2588 \par
2589 \unskip
2590 \ifvoid\@mpfootins\else
2591 \l@dunboxmpfoot
2592 \fi
2593 \l@dfetendmini%
2594 }
2595

```

29 Indexing

Here's some code for indexing using page & line numbers.

`\pagelinesep` In order to get a correct line number we have to use the label/ref mechanism. These
`\edindexlab` macros are for that.

```

\c@labidx 2596 \newcommand{\pagelinesep}{-}
2597 \newcommand{\edindexlab}{\&}
2598 \newcounter{labidx}
2599 \setcounter{labidx}{0}
2600

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

2601 \newcommand{\doedindexlabel}{\stepcounter{labidx}%
2602 \edlabel{\edindexlab\thelabidx}}
2603

```

`\thepageline` This macro makes up the page/line number combo from the label/ref.

```

2604 \newcommand{\thepageline}{%
2605 \thepage\pagelinesep\lineref{\edindexlab\thelabidx}}
2606

```

The memoir class provides more flexible indexing than the standard classes. We need different code if the memoir class is being used.

```
2607 \ifclassloaded{memoir}{%
```

```
memoir is being used.
```

`\makeindex` Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing. In this case `\edindex` has an optional argument. We use the hook provided in memoir v1.61.

```
2608 \g@addto@macro{\makememindexhook}{%
2609   \def\edindex{\@bsphack%
2610     \ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
2611   \newcommand{\edindex}[2][\jobname]{\@bsphack\esphack}
```

`\l@d@index` `\l@d@index[file]` is the first stage of `\edindex`, handling the idx file. This is a virtually a verbatim copy of memoir's `\@index`, the change being calling `\l@dwrindexm@m` instead of `\@wrindexm@m`.

```
2612 \def\l@d@index[#1]{%
2613   \ifundefined{#1idxfile}%
2614   {\ifreportnoidxfile
2615     \led@warn@NoIndexFile{#1}%
2616     \fi
2617     \begingroup
2618     \@sanitize
2619     \@nowrindex}%
2620   {\def\@idxfile{#1}%
2621     \doedindexlabel
2622     \begingroup
2623     \@sanitize
2624     \l@d@wrindexm@m}}
```

`\l@d@wrindexm@m` `\l@d@wrindexm@m{item}` writes the idx file name and the indexed item to the aux file. These are almost verbatim copies of memoir's `\@wrindexm@m` and `\@@wrindexhyp`.

```
2625 \newcommand{\l@d@wrindexm@m}[1]{\l@d@wrindexhyp#1||\%}
2626 \def\l@d@wrindexhyp#1|#2|#3\%{
2627   \ifshowindexmark\@showidx{#1}\fi
2628   \ifx\#2\%
2629     \protected@write\@auxout{%
2630       {\string\@@wrindexm@m{\@idxfile}{#1|hyperpage}{\thepageline}}%
2631     \else
2632       \def\Hy@temp@A{#2}%
2633       \ifx\Hy@temp@A\HyInd@ParenLeft
2634         \protected@write\@auxout{%
2635           {\string\@@wrindexm@m{\@idxfile}{#1|#2hyperpage}{\thepageline}}%
2636         \else
2637           \protected@write\@auxout{%
2638             {\string\@@wrindexm@m{\@idxfile}{#1|#2}{\thepageline}}%
2639           \fi
```



```

2640     \fi
2641     \endgroup
2642     \@esphack}

```

That finishes the memoir-specific code.

```
2643 }{%
```

memoir is not being used, which makes life somewhat simpler.

`\makeindex` Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to `\edindex` do nothing.

```

2644 \g@addto@macro{\makeindex}{%
2645     \def\edindex{\@bsphack
2646         \doedindexlabel
2647         \begingroup
2648         \@sanitize
2649         \wredindex}}
2650 \newcommand{\edindex}[1]{\@bsphack\@esphack}

```

`\@wredindex` Write the index information to the `idx` file.

```

2651 \newcommand{\@wredindex}[1]{%
2652     \protected@write\@indexfile{}%
2653         {\string\indexentry{#1}{\thepageline}}%
2654     \endgroup
2655     \@esphack}

```

That finishes the non-memoir index code.

```

2656 }
2657

```

`\l@d@wrindexhyp` If the `hyperref` package is not loaded, it doesn't make sense to clutter up the index with hyperreffing things.

```

2658 \AtBeginDocument{\@ifpackageloaded{hyperref}{}{%
2659     \def\l@d@wrindexhyp#1||\{\%
2660         \ifshowindexmark\@showidx{#1}\fi
2661         \protected@write\@auxout{}%
2662             {\string\@wrindexm@m{\@idxfile}{#1}{\thepageline}}%
2663         \endgroup
2664         \@esphack}}
2665

```

30 Macro as environment

The following is borrowed, and renamed, from the `amsmath` package. See also the CTT thread 'eq and amstex', 1995/08/31, started by Keith Reckdahl and ended definitively by David M. Jones.

Several of the `[math]` macros scan their body twice. This means we must collect all text in the body of an environment form before calling the macro.

`\@emptytoks` This is actually defined in the `amsgen` package.

```
2666 \newtoks\@emptytoks
2667
```

The rest is from `amsmath`.

`\l@denvbody` A token register to contain the body.

```
2668 \newtoks\l@denvbody
2669
```

`\addtol@denvbody` `\addtol@denvbody{arg}` adds `arg` to the token register `\l@denvbody`.

```
2670 \newcommand{\addtol@denvbody}[1]{%
2671   \global\l@denvbody\expandafter{\the\l@denvbody#1}}
2672
```

`\l@dcollect@body` The macro `\l@dcollect@body` starts the scan for the `\end{...}` command of the current environment. It takes a macro name as argument. This macro is supposed to take the whole body of the environment as its argument. For example, given `cenv#1{...}` as a macro that processes `#1`, then the environment form, `\begin{env}` would call `\l@dcollect@body\cenv`.

```
2673 \newcommand{\l@dcollect@body}[1]{%
2674   \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}%
2675   \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currenvir}}%
2676   \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
2677   \begingroup
2678     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
2679     \edef\processl@denvbody{\expandafter\noexpand\csname\@currenvir\endcsname}%
2680     \processl@denvbody}
2681
```

`\l@dpush@begins` When adding a piece of the current environment's contents to `\l@denvbody`, we scan it to check for additional `\begin` tokens, and add a 'b' to the stack for any that we find.

```
2682 \def\l@dpush@begins#1\begin#2{%
2683   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
2684
```

`\l@dcollect@@body` `\l@dcollect@@body` takes two arguments: the first will consist of all text up to the next `\end` command, and the second will be the `\end` command's argument. If there are any extra `\begin` commands in the body text, a marker is pushed onto a stack by the `\l@dpush@begins` function. Empty state for this stack means we have reached the `\end` that matches our original `\begin`. Otherwise we need to include the `\end` and its argument in the material we are adding to the environment body accumulator.

```
2685 \def\l@dcollect@@body#1\end#2{%
2686   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
2687     \expandafter\@gobble\l@dbegin@stack}%
2688   \ifx\@empty\l@dbegin@stack
```

```

2689 \endgroup
2690 \@checkend{#2}%
2691 \addtol@denbody{#1}%
2692 \else
2693 \addtol@denbody{#1\end{#2}}%
2694 \fi
2695 \processl@denbody % A little tricky! Note the grouping
2696 }
2697

```

There was a question on CTT about how to use `\collect@body` for a macro taking an argument. The following is part of that thread.

```

From: Heiko Oberdiek <oberdiek@uni-freiburg.de>
Newsgroups: comp.text.tex
Subject: Re: Using \collect@body with commands that take >1 argument
Date: Fri, 08 Aug 2003 09:03:20 +0200

```

```

eed132@psu.edu (Evan) wrote:
> I'm trying to make a new Latex environment that acts like the>
> \colorbox command that is part of the color package. I looked through
> the FAQ and ran across this bit about using the \collect@body command
> that is part of AMSLaTeX:
> http://www.tex.ac.uk/cgi-bin/texfaq2html?label=cmdasenv
>
> It almost works. If I do something like the following:
> \newcommand{\redbox}[1]{\colorbox{red}{#1}}
>
> \makeatletter
> \newenvironment{redbox}{\collect@body \redbox}{\}

```

You will get an error message: Command `\redbox` already defined.
Thus you must rename either the command `\redbox` or the environment name.

```

> \begin{coloredbox}{blue}
> Yadda yadda yadda... this is on a blue background...
> \end{coloredbox}
> and can't figure out how to make the \collect@body take this.

> \collect@body \colorbox{red}
> \collect@body {\colorbox{red}}

```

The argument of `\collect@body` has to be one token exactly.

```

\documentclass{article}
\usepackage{color}
\usepackage{amsmath}

\newcommand{\redbox}[1]{\colorbox{red}{#1}}

```

```

\makeatletter
\newenvironment{coloredbox}[1]{%
  \def\next@{\colorbox{#1}}%
  \collect@body\next@
}{%}

% ignore spaces at begin and end of environment
\newenvironment{coloredboxII}[1]{%
  \def\next@{\mycoloredbox{#1}}%
  \collect@body\next@
}{%}
\newcommand{\mycoloredbox}[2]{%
  \colorbox{#1}{\ignorespaces#2\unskip}%
}

% support of optional color model argument
\newcommand\coloredboxIII\endcsname{}
\def\coloredboxIII#1{%
  \@coloredboxIII{#1}%
}
\def\@coloredboxIII#1#2{%
  \def\next@{\mycoloredboxIII{#1}{#2}}%
  \collect@body\next@
}
\newcommand{\mycoloredboxIII}[3]{%
  \colorbox#1{#2}{\ignorespaces#3\unskip}%
}

\makeatother

\begin{document}
  Black text before
  \begin{coloredbox}{blue}
    Hello World
  \end{coloredbox}
  Black text after

  Black text before
  \begin{coloredboxII}{blue}
    Hello World
  \end{coloredboxII}
  Black text after

  Black text before
  \begin{coloredboxIII}[rgb]{0,0,1}
    Hello World
  \end{coloredboxIII}
  Black text after

\end{document}

```

Yours sincerely
Heiko <oberdiek@uni-freiburg.de>

31 Verse

This is principally Wayne Sullivan's code and commentary from EDSTANZA [Sul92].

The macro `\hangingsymbol` is used to insert a symbol on each hanging of verses. For example, in french typographie the symbol is '['. We obtain it by the next code:

```
\renewcommand{\hangingsymbol}{[,]}
```

The `\ifinstanza` boolean is used to be sure that we are in a stanza part.

```
\hangingsymbol
\ifinstanza 2698 \newcommand*{\hangingsymbol}{}
2699 \newif\ifinstanza
2700 \instanzafalse
```

`\inserthangingsymbol` After each insertion of true line, the value of `\line@num` counter is put into the `\prevline@num` counter. When we change verse, the `\line@num` is not equal to the `\prevline@num`. But if a verse is in two ore more lines, the second line (and next) has the same number as the first line of the verse. Consequently, the `\line@num` is equal to the `\prevline@num`. This property is used by the `\inserthangingsymbol` command to insert the hanging symbol at each verse's hanging.

```
2701 \newcount\prevline@num
2702 \newcommand{\inserthangingsymbol}{\ifinstanza\ifnum\line@num=\prevline@num\hfill\hangingsymbol\fi\fi}
```

`\ampersand` Within a stanza the `\&` macro is going to be usurped. We need an alias in case an `&` needs to be typeset in a stanza. Define it rather than letting it in case some other package has already defined it.

```
2703 \newcommand*{\ampersand}{\char'\&}
2704
```

`\stanza@count` Before we can define the main macros we need to save and reset some category codes. To save the current values we use `\next` and `\body` from the `\loop` macro.

```
2705 \chardef\body=\catcode'\@
2706 \catcode'\@=11
2707 \chardef\next=\catcode'\&
2708 \catcode'\&=\active
2709
```

A count register is allocated for counting lines in a stanza; also allocated is a dimension register which is used to specify the base value for line indentation; all stanza indentations are multiples of this value. The default value of `\stanzaindentbase` is 20pt.

```

2710 \newcount\stanza@count
2711 \newlength{\stanzaindentbase}
2712 \setlength{\stanzaindentbase}{20pt}
2713

```

`\strip@szacnt` The indentations of stanza lines are non-negative integer multiples of the unit called `\stanzaindentbase`. To make it easier for the user to specify these numbers, some list macros are defined. These take numerical values in a list separated by commas and assign the values to special control sequences using `\mathchardef`. Though this does limit the range from 0 to 32767, it should suffice for most applications, including *penalties*, which will be discussed below.

```

2714 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
2715 \newcommand*\setstanzavalues}[2]{\def\@tempa{#2,,|}%
2716     \stanza@count\z@
2717     \def\next{\expandafter\strip@szacnt\@tempa
2718         \ifx\@tempb\empty\let\next\relax\else
2719         \expandafter\mathchardef\csname #1@\number\stanza@count
2720         \@endcsname\@tempb\relax
2721         \advance\stanza@count\@ne\fi\next}%
2722     \next}
2723

```

`\setstanzaindents` In the original `\setstanzavalues{sza}{...}` had to be called to set the indents, and similarly `\setstanzavalues{szp}{...}` to set the penalties. These two macros are a convenience to give the user one less thing to worry about (misspelling the first argument). Since version 0.13, the `stanzaindentsrepetition` counter can be used when the indentation is repeated every *n* verses. The `\managestanza@modulo` is a command which modifies the counter `stanza@modulo`. The command adds 1 to `stanza@modulo`, but if `stanza@modulo` is equal to the `stanzaindentsrepetition` counter, the command restarts it.

```

2724 \newcommand*\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
2725 \newcommand*\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
2726
2727 \newcounter{stanzaindentsrepetition}
2728 \newcount\stanza@modulo
2729
2730 \newcommand*\managestanza@modulo}[0]{
2731     \advance\stanza@modulo\@ne
2732     \ifnum\stanza@modulo>\value{stanzaindentsrepetition}
2733         \stanza@modulo\@ne
2734     \fi
2735 }

```

`\stanza@line` Now we arrive at the main works. `\stanza@line` sets the indentation for the line and starts a numbered paragraph—each line is treated as a paragraph.

`\stanza@hang` `\stanza@hang` sets the hanging indentation to be used if the stanza line requires more than one print line. If it is known that each stanza line will fit on one print line, it is advisable to set the hanging indentation to zero. `\sza@penalty` places

the specified penalty following each stanza line. By default, this facility is turned off so that no penalty is included. However, the user may initiate these penalties to indicate good and bad places in the stanza for page breaking.

```

2736 \def\stanza@line{
2737   \ifnum\value{stanzaindentrepetition}=0
2738     \parindent=\csname sza@\number\stanza@count
2739       @\endcsname\stanzaindentbase
2740   \else
2741     \managestanza@modulo
2742     \parindent=\csname sza@\number\stanza@modulo
2743       @\endcsname\stanzaindentbase
2744   \fi
2745   \pstart\stanza@hang\ignorespaces}
2746 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
2747   \hangindent\expandafter
2748   \noexpand\csname sza@0@\endcsname\stanzaindentbase
2749   \hangafter\@ne}
2750 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
2751   \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
2752   \penalty\fi\count@}

```

`\startstanzahook` Now we have the components of the `\stanza` macro, which appears at the start of a group of lines. This macro initializes the count and checks to see if hanging indentation and penalties are to be included. Hanging indentation suspends the line count, so that the enumeration is by verse line rather than by print line. If the print line count is desired, invoke `\let\startlock=\relax` and do the same for `\endlock`. Here and above we have used `\xdef` to make the stored macros take up a bit less space, but it also makes them more obscure to the reader. Lines of the stanza are delimited by ampersands `&`. The last line of the stanza must end with `&`. For convenience the macro `\endstanzaextra` is included. The user may use this to add vertical space or penalties between stanzas.

As a further convenience, the macro `\startstanzahook` is called at the beginning of a stanza. This can be defined to do something useful.

```

2753 \let\startstanzahook\relax
2754 \let\endstanzaextra\relax
2755 \xdef\stanza{\noexpand\instanzatrue\expandafter
2756   \begingroup\startstanzahook%
2757   \catcode'\&\active\global\stanza@count\@ne\stanza@modulo\@ne
2758   \noexpand\ifnum\expandafter\noexpand
2759   \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
2760   \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
2761   \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
2762   \expandafter\noexpand\csname szp@0@\endcsname=\z@
2763   \let\noexpand\sza@penalty\relax\noexpand\fi \def\noexpand&{%
2764   \noexpand\endlock\noexpand\pend\noexpand\sza@penalty\global
2765   \advance\stanza@count\@ne\noexpand\stanza@line}\def\noexpand
2766   &{\noexpand\endlock\noexpand\pend\endgroup\noexpand\instanzafalse\expandafter\endstanzaextra
2767   \noexpand\stanza@line}

```

2768

`\flagstanza` Use `\flagstanza[len]{text}` at the start of a line to put text a distance *len* before the start of the line. The default for *len* is `\stanzaindentbase`.

```
2769 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
2770   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
2771
```

The ampersand & is used to mark the end of each stanza line, except the last, which is marked with `\&`. This means that `\halign` may not be used directly within a stanza line. This does not affect macros involving alignments defined outside `\stanza \&`. Since these macros usurp the control sequence `\&`, the replacement `\ampersand` is defined to be used if this symbol is needed in a stanza. Also we reset the modified category codes and initialize the penalty default.

```
2772 \catcode'\&=\next
2773 \catcode'\@=\body
2774 %% \let\ampersand=\&
2775 \setstanzavalues{szp}{0}
2776
```

32 Arrays and tables

This is based on the work by Herbert Breger in developing `tabmac.tex`.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This is file tabmac.tex 1.0.
% You find here macros for tabular structures compatible with
% Edmac (authored by Lavagnino/Wujastyk). The use of the macros is
% explained in German language in file tabanlei.dvi. The macros were
% developed for Edmac 2.3, but this file has been adjusted to Edmac 3.16.
%
% ATTENTION: This file uses some Edmac control sequences (like
% \text, \footnote etc.) and redefines \morenoexpands. If you yourself
% redefined some Edmac control sequences, be careful: some adjustments
% might be necessary.
% October 1996
%
% My kind thanks to Nora G^deke for valuable support. Any hints and
% comments are welcome, please contact Herbert Breger,
% Leibniz-Archiv, Waterloostr. 8, D -- 30169 Hannover, Germany
% Tel.: 511 - 1267 327
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The original `tabmac.tex` file was void of comments or any explanatory text other than the above notice. The algorithm is Breger's. I have made some cosmetic changes to the original code and reimplemented some things so they are more LaTeX-like. All the commentary is mine, as are any mistakes or errors.

`\l@dtabnoexpands` An extended and modified version of the original additional no expansions..

```

2777 \newcommand*{\l@dtabnoexpands}{%
2778   \def\ss{\noexpand\ss}%
2779   \def\"##1{\noexpand\"##1}%
2780   \def\'##1{\noexpand\'##1}%
2781   \def\`##1{\noexpand\`##1}%
2782   \def\^##1{\noexpand\^##1}%
2783   \def\phantom##1{\noexpand\phantom{##1}}%
2784   \def\hphantom##1{\noexpand\hphantom{##1}}%
2785   \def\underbrace##1{\noexpand\underbrace{##1}}%
2786   \def\dots{\noexpand\dots}%
2787   \let\rtab=0%
2788   \let\ctab=0%
2789   \let\ltab=0%
2790   \let\rtabtext=0%
2791   \let\ltabtext=0%
2792   \let\ctabtext=0%
2793   \let\edbeforetab=0%
2794   \let\edaftertab=0%
2795   \let\edatab=0%
2796   \let\edatabell=0%
2797   \let\edatleft=0%
2798   \let\edatright=0%
2799   \let\edvertline=0%
2800   \let\edvertdots=0%
2801   \let\edrowfill=0%
2802 }
2803

```

`\l@dampcount` `\l@dampcount` is a counter for the & column dividers and `\l@dcolcount` is a counter for the columns. These were `\Undcount` and `\stellencount` respectively.

```

2804 \newcount\l@dampcount
2805 \l@dampcount=1\relax
2806 \newcount\l@dcolcount
2807 \l@dcolcount=0\relax
2808

```

`\hilfsbox` Some (temporary) helper items.

```

\hilfsskip 2809 \newbox\hilfsbox
\Hilfsbox 2810 \newskip\hilfsskip
\hilfscount 2811 \newbox\Hilfsbox
2812 \newcount\hilfscount
2813

```

30 columns should be adequate (compared to the original 60). These are the column widths. (Originally these were German spelled numbers e.g., `\eins`, `\zwei`, etc).

```

2814 \newdimen\dcoli
2815 \newdimen\dcolii

```

```

2816 \newdimen\dcoli
2817 \newdimen\dcoliv
2818 \newdimen\dcolv
2819 \newdimen\dcolvi
2820 \newdimen\dcolvii
2821 \newdimen\dcolviii
2822 \newdimen\dcolix
2823 \newdimen\dcolx
2824 \newdimen\dcolxi
2825 \newdimen\dcolxii
2826 \newdimen\dcolxiii
2827 \newdimen\dcolxiv
2828 \newdimen\dcolxv
2829 \newdimen\dcolxvi
2830 \newdimen\dcolxvii
2831 \newdimen\dcolxviii
2832 \newdimen\dcolxix
2833 \newdimen\dcolxx
2834 \newdimen\dcolxxi
2835 \newdimen\dcolxxii
2836 \newdimen\dcolxxiii
2837 \newdimen\dcolxxiv
2838 \newdimen\dcolxxv
2839 \newdimen\dcolxxvi
2840 \newdimen\dcolxxvii
2841 \newdimen\dcolxxviii
2842 \newdimen\dcolxxix
2843 \newdimen\dcolxxx
2844 \newdimen\dcolerr % added for error handling
2845

```

`\l@dcolwidth` This is a cunning way of storing the columnwidths indexed by the column number `\l@dcolcount`, like an array. (was `\Dimenzuordnung`)

```

2846 \newcommand{\l@dcolwidth}{\ifcase \the\l@dcolcount \dcoli %???
2847 \or \dcoli \or \dcolii \or \dcoliii
2848 \or \dcoliv \or \dcolv \or \dcolvi
2849 \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
2850 \or \dcolxi \or \dcolxii \or \dcolxiii
2851 \or \dcolxiv \or \dcolxv \or \dcolxvi
2852 \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
2853 \or \dcolxxi \or \dcolxxii \or \dcolxxiii
2854 \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
2855 \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
2856 \else \dcolerr \fi}
2857

```

`\step1@dcolcount` This increments the column counter, and issues an error message if it is too large.

```

2858 \newcommand*{\step1@dcolcount}{\advance\l@dcolcount\@ne
2859 \ifnum\l@dcolcount>30\relax
2860 \led@err@TooManyColumns

```

```
2861 \fi}
2862
```

`\l@dssetmaxcolwidth` Sets the column width to the maximum value seen so far. (was `\dimenzuordnung`)

```
2863 \newcommand{\l@dssetmaxcolwidth}{%
2864   \ifdim\l@dcolwidth < \wd\hilfsbox
2865     \l@dcolwidth = \wd\hilfsbox
2866   \else \relax \fi}
2867
```

`\EDTEXT` We need to be able to modify the `\edtext` and `\critext` macros and also restore `\xedtext` their original definitions.

```
\CRITEXT 2868 \let\EDTEXT=\edtext
\xcritext 2869 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
2870 \let\CRITEXT=\critext
2871 \long\def\xcritext #1#2/{\CRITEXT{#1}{#2}/}
```

`\EDLABEL` We need to be able to modify and restore the `\edlabel` macro.

```
\xedlabel 2872 \let\EDLABEL=\edlabel
2873 \newcommand*{\xedlabel}[1]{\EDLABEL{#1}}
```

`\EDINDEX` Macros supporting modification and restoration of `\edindex`.

```
\xedindex 2874 \let\EDINDEX=\edindex
\nulledindex 2875 \ifl@dmemoir
2876   \newcommand{\xedindex}{\@bsphack%
2877     \@ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
2878   \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
2879 \else
2880   \newcommand{\xedindex}{\@bsphack%
2881     \doedindexlabel
2882     \begingroup
2883     \@sanitize
2884     \wredindex}
2885   \newcommand{\nulledindex}[1]{\@bsphack\@esphack}
2886 \fi
2887
```

`\A@@footnote` We need to be able to modify `ledmac's` footnote macros and restore their original `\B@@footnote` definitions. There are five of these.

```
\C@@footnote 2888 \let\A@@footnote=\Afootnote
\D@@footnote 2889 \let\B@@footnote=\Bfootnote
\E@@footnote 2890 \let\C@@footnote=\Cfootnote
2891 \let\D@@footnote=\Dfootnote
2892 \let\E@@footnote=\Efootnote
```

`\@line@num` Macro supporting restoration of `\linenum`.

```
2893 \let\@line@num=\linenum
```

`\l@dgobbledarg` `\l@dgobbledarg` replaces its delineated argument by `\relax` (was `\verschwinden`).
`\l@dgobblearg` `\l@dgobblearg{<arg>}` replaces its argument by `\relax`.

```
2894 \def\l@dgobbledarg #1/{\relax}
2895 \newcommand*{\l@dgobblearg}[1]{\relax}
2896
```

```
\Relax
\NEXT 2897 \let\Relax=\relax
\@hilfs@count 2898 \let\NEXT=\next
2899 \newcount\@hilfs@count
2900
```

`\measuremcell` Measure (recursively) the width required for a math cell. (was `\messen`)

```
2901 \def\measuremcell #1{%
2902   \ifx #1\ \ifnum\l@dc@count=0\let\NEXT\relax%
2903   \else\l@dcheckcols%
2904   \l@dc@count=0%
2905   \let\NEXT\measuremcell%
2906   \fi%
2907   \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
2908   \step\l@dc@count%
2909   \l@dsetmaxcolwidth%
2910   \let\NEXT\measuremcell%
2911   \fi\NEXT}
2912
```

`\measuretcell` Measure (recursively) the width required for a text cell. (was `\messentext`)

```
2913 \def\measuretcell #1{%
2914   \ifx #1\ \ifnum\l@dc@count=0\let\NEXT\relax%
2915   \else\l@dcheckcols%
2916   \l@dc@count=0%
2917   \let\NEXT\measuretcell%
2918   \fi%
2919   \else\setbox\hilfsbox=\hbox{#1}%
2920   \step\l@dc@count%
2921   \l@dsetmaxcolwidth%
2922   \let\NEXT\measuretcell%
2923   \fi\NEXT}
2924
```

`\measuremrow` Measure (recursively) the width required for a math row. (was `\Messen`)

```
2925 \def\measuremrow #1\{%
2926   \ifx #1\let\NEXT\relax%
2927   \else\measuremcell #1\&\&\&%
2928   \let\NEXT\measuremrow%
2929   \fi\NEXT}
```

`\measuretrow` Measure (recursively) the width required for a text row. (was `\Messentext`)

```
2930 \def\measuretrow #1\{%
```

```

2931 \ifx #1&\let\NEXT\relax%
2932 \else\measurecell #1&\&\&%
2933 \let\NEXT\measuretrow%
2934 \fi\NEXT}
2935

```

`\edtabcolsep` The length `\edtabcolsep` controls the distance between columns. (was `\abstand`)

```

2936 \newskip\edtabcolsep
2937 \global\edtabcolsep=10pt
2938

```

`\NEXT`

```

\Next 2939 \let\NEXT\relax
2940 \let\Next=\next

```

`\variab`

```

2941 \newcommand{\variab}{\relax}
2942

```

`\l@dcheckcols` Check that the number of columns is consistent. (was `\tabfehlermeldung`)

```

2943 \newcommand*{\l@dcheckcols}{%
2944 \ifnum\l@dcolcount=1\relax
2945 \else
2946 \ifnum\l@dampcount=1\relax
2947 \else
2948 \ifnum\l@dcolcount=\l@dampcount\relax
2949 \else
2950 \l@d@err@UnequalColumns
2951 \fi
2952 \fi
2953 \l@dampcount=\l@dcolcount
2954 \fi}
2955

```

`\l@dmodforcritext` Modify and restore various macros for when `\critext` is used.

```

\l@drestoreforcritext 2956 \newcommand{\l@dmodforcritext}{%
2957 \let\critext\relax%
2958 \let\Afootnote\l@dgobbledarg%
2959 \let\Bfootnote\l@dgobbledarg%
2960 \let\Cfootnote\l@dgobbledarg%
2961 \let\Dfootnote\l@dgobbledarg%
2962 \let\Efootnote\l@dgobbledarg%
2963 \let\edindex\nulledindex%
2964 \let\linenum@gobble}
2965 \newcommand{\l@drestoreforcritext}{%
2966 \def\Afootnote##1##2/{\A@footnote{##1}{##2}}%
2967 \def\Bfootnote##1##2/{\B@footnote{##1}{##2}}%
2968 \def\Cfootnote##1##2/{\C@footnote{##1}{##2}}%
2969 \def\Dfootnote##1##2/{\D@footnote{##1}{##2}}%

```

```

2970 \def\Efootnote##1##2/{\E@@footnote{##1}{##2}}%
2971 \let\edindex\xedndex}
2972

```

\l@dmodforedtext Modify and restore various macros for when \edtext is used.

```

\l@drestoreforedtext 2973 \newcommand{\l@dmodforedtext}{%
2974 \let\edtext\relax
2975 \let\Afootnote\l@dgobblearg
2976 \let\Bfootnote\l@dgobblearg
2977 \let\Cfootnote\l@dgobblearg
2978 \let\Dfootnote\l@dgobblearg
2979 \let\Efootnote\l@dgobblearg
2980 \let\edindex\nulledindex
2981 \let\linenum@gobble}
2982 \newcommand{\l@drestoreforedtext}{%
2983 \def\Afootnote##1{\A@@footnote{##1}}%
2984 \def\Bfootnote##1{\B@@footnote{##1}}%
2985 \def\Cfootnote##1{\C@@footnote{##1}}%
2986 \def\Dfootnote##1{\D@@footnote{##1}}%
2987 \def\Efootnote##1{\E@@footnote{##1}}%
2988 \let\edindex\xedndex}
2989

```

\l@dnullfills Nullify and restore some column fillers, etc.

```

\l@drestorefills 2990 \newcommand{\l@dnullfills}{%
2991 \def\edlabel##1{}%
2992 \def\edrowfill##1##2##3{}%
2993 }
2994 \newcommand{\l@drestorefills}{%
2995 \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
2996 }
2997

```

The original definition of \rverteilen and friends ('verteilen' is approximately 'distribute') was along the lines:

```

\def\rverteilen #1&{\def\label##1{%
\ifx #1! \ifnum\l@dcolcount=0%\removelastskip
\let\Next\relax%
\else\l@dcolcount=0%
\let\Next=\rverteilen%
\fi%
\else%
\footnoteverschw%
\stepl@dcolcount%
\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
\let\critext=\xcritext\let\Dfootnote=D@@footnote
\let\Afootnote=A@@footnote\let\Bfootnote=B@@footnote
\let\Cfootnote=C@@footnote\let\linenum=\@line@num%
\hilfsskip=Dimenzuordnung%

```

```

\advance\hilfsskip by -\wd\hilfsbox
\def\label##1{\xlabel{##1}}%
\hskip\hilfsskip$\displaystyle{#1}$%
\hskip\edtabcolsep%
\let\Next=\rverteilen%
\fi\Next}

```

where the lines

```

\let\critext=\xcritext\let\Dfootnote=\D@@footnote
\let\Afootnote=\A@@footnote\let\Bfootnote=\B@@footnote
\let\Cfootnote=\C@@footnote\let\linenum=\@line@num%
\hilfsskip=\Dimenzuordnung%
\advance\hilfsskip by -\wd\hilfsbox
\def\label##1{\xlabel{##1}}%

```

were common across the several **verteilen** macros, and also

```

\def\footnoteverschw{%
\let\critext\relax
\let\Afootnote=\verschwinden
\let\Bfootnote=\verschwinden
\let\Cfootnote=\verschwinden
\let\Dfootnote=\verschwinden
\let\linenum=\@gobble}

```

\letsforverteilen Gathers some lets and other code that is common to the **verteilen** macros.

```

2998 \newcommand{\letsforverteilen}{%
2999 \let\critext\xcritext
3000 \let\edtext\xedtext
3001 \let\edindex\xedindex
3002 \let\Afootnote\A@@footnote
3003 \let\Bfootnote\B@@footnote
3004 \let\Cfootnote\C@@footnote
3005 \let\Dfootnote\D@@footnote
3006 \let\Efootnote\E@@footnote
3007 \let\linenum\@line@num
3008 \hilfsskip=\l@dcwidth%
3009 \advance\hilfsskip by -\wd\hilfsbox
3010 \def\edlabel##1{\xedlabel{##1}}
3011

```

\setmcellright Typeset (recursively) cells of display math right justified. (was *\rverteilen*)

```

3012 \def\setmcellright #1{\def\edlabel##1{%
3013 \let\edindex\nulledindex
3014 \ifx #1\ \ifnum\l@dcwidth=0%\removelastskip
3015 \let\Next\relax%
3016 \else\l@dcwidth=0%

```

```

3017             \let\Next=\setmcellright%
3018         \fi%
3019     \else%
3020         \disablel@dtabfeet%
3021         \step1@dcolcount%
3022         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
3023         \letsforverteilen%
3024         \hskip\hilfsskip$\displaystyle{#1}$%
3025         \hskip\edtabcolsep%
3026         \let\Next=\setmcellright%
3027     \fi\Next}
3028

```

`\settccllright` Typeset (recursively) cells of text right justified. (was `\rverteilentext`)

```

3029 \def\settccllright #1&{\def\edlabel##1{}%
3030             \let\edindex\nulledindex
3031     \ifx #1\\ \ifnum\l@dcolcount=0\removelastskip
3032             \let\Next\relax%
3033     \else\l@dcolcount=0%
3034             \let\Next=\settccllright%
3035     \fi%
3036 \else%
3037     \disablel@dtabfeet%
3038     \step1@dcolcount%
3039     \setbox\hilfsbox=\hbox{#1}%
3040     \letsforverteilen%
3041     \hskip\hilfsskip#1%
3042     \hskip\edtabcolsep%
3043     \let\Next=\settccllright%
3044 \fi\Next}

```

`\setmcellleft` Typeset (recursively) cells of display math left justified. (was `\lverteilen`)

```

3045 \def\setmcellleft #1&{\def\edlabel##1{}%
3046             \let\edindex\nulledindex
3047     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
3048     \else\l@dcolcount=0%
3049             \let\Next=\setmcellleft%
3050     \fi%
3051 \else \disablel@dtabfeet%
3052     \step1@dcolcount%
3053     \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
3054     \letsforverteilen
3055     $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
3056     \let\Next=\setmcellleft%
3057 \fi\Next}
3058

```

`\settccllleft` Typeset (recursively) cells of text left justified. (was `\lverteilentext`)

```

3059 \def\settccllleft #1&{\def\edlabel##1{}%

```



```

3060             \let\edindex\nulledindex
3061 \ifx #1\ \ifnum\l@dc@count=0 \let\Next\relax%
3062             \else\l@dc@count=0%
3063             \let\Next=\settc@left%
3064             \fi%
3065 \else \disabl@dtabfeet%
3066         \stepl@dc@count%
3067         \setbox\hilfsbox=\hbox{#1}%
3068         \letsforverteilen
3069         #1\hskip\hilfsskip\hskip\edtabcolsep%
3070         \let\Next=\settc@left%
3071 \fi\Next}

```

`\setmcellcenter` Typeset (recursively) cells of display math centered. (was `\zverteilen`)

```

3072 \def\setmcellcenter #1{\def\edlabel##1{%
3073             \let\edindex\nulledindex
3074 \ifx #1\ \ifnum\l@dc@count=0 \let\Next\relax%
3075             \else\l@dc@count=0%
3076             \let\Next=\setmcellcenter%
3077             \fi%
3078 \else \disabl@dtabfeet%
3079         \stepl@dc@count%
3080         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
3081         \letsforverteilen%
3082         \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
3083         \hskip\edtabcolsep%
3084         \let\Next=\setmcellcenter%
3085 \fi\Next}
3086

```

`\settc@center` Typeset (recursively) cells of text centered. (new)

```

3087 \def\settc@center #1{\def\edlabel##1{%
3088             \let\edindex\nulledindex
3089 \ifx #1\ \ifnum\l@dc@count=0 \let\Next\relax%
3090             \else\l@dc@count=0%
3091             \let\Next=\settc@center%
3092             \fi%
3093 \else \disabl@dtabfeet%
3094         \stepl@dc@count%
3095         \setbox\hilfsbox=\hbox{#1}%
3096         \letsforverteilen%
3097         \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
3098         \hskip\edtabcolsep%
3099         \let\Next=\settc@center%
3100 \fi\Next}
3101

```

`\NEXT`

```

3102 \let\NEXT=\relax
3103

```

`\setmrowright` Typeset (recursively) rows of right justified math. (was `\rsetzen`)

```
3104 \def\setmrowright #1\\{%
3105     \ifx #1& \let\NEXT\relax
3106     \else \centerline{\setmcellright #1&\\&\\&}
3107         \let\NEXT=\setmrowright
3108     \fi\NEXT}
```

`\settroright` Typeset (recursively) rows of right justified text. (was `\rsetzentext`)

```
3109 \def\settroright #1\\{%
3110     \ifx #1& \let\NEXT\relax
3111     \else \centerline{\settrcellright #1&\\&\\&}
3112         \let\NEXT=\settroright
3113     \fi\NEXT}
3114
```

`\setmrowleft` Typeset (recursively) rows of left justified math. (was `\lsetzen`)

```
3115 \def\setmrowleft #1\\{%
3116     \ifx #1& \let\NEXT\relax
3117     \else \centerline{\setmcellleft #1&\\&\\&}
3118         \let\NEXT=\setmrowleft
3119     \fi\NEXT}
```

`\settrorleft` Typeset (recursively) rows of left justified text. (was `\lsetzentext`)

```
3120 \def\settrorleft #1\\{%
3121     \ifx #1& \let\NEXT\relax
3122     \else \centerline{\settrcellleft #1&\\&\\&}
3123         \let\NEXT=\settrorleft
3124     \fi\NEXT}
3125
```

`\setmrowcenter` Typeset (recursively) rows of centered math. (was `\zsetzen`)

```
3126 \def\setmrowcenter #1\\{%
3127     \ifx #1& \let\NEXT\relax%
3128     \else \centerline{\setmcellcenter #1&\\&\\&}
3129         \let\NEXT=\setmrowcenter
3130     \fi\NEXT}
```

`\settrorcenter` Typeset (recursively) rows of centered text. (new)

```
3131 \def\settrorcenter #1\\{%
3132     \ifx #1& \let\NEXT\relax
3133     \else \centerline{\settrcellcenter #1&\\&\\&}
3134         \let\NEXT=\settrorcenter
3135     \fi\NEXT}
3136
```

`\nullsetzen` (was `\nullsetzen`)

```
3137 \newcommand{\nullsetzen}{%
3138     \step1@dcolcount%
3139     \l@dcolwidth=0pt%
```

```

3140 \ifnum\l@dcolcount=30\let\NEXT\relax%
3141 \l@dcolcount=0\relax
3142 \else\let\NEXT\nullsetzen%
3143 \fi\NEXT}
3144

\edatleft \edatleft[ $]{\langle symbol \rangle}{\langle len \rangle}$  (combination and generalisation of original
\Seklam and \Seklamgl). Left  $\langle symbol \rangle$ ,  $2\langle len \rangle$  high with prepended  $\langle math \rangle$  verti-
cally centered.

3145 \newcommand{\edatleft}[3][\@empty]{%
3146 \ifx#1\@empty
3147 \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
3148 depth 0pt \right. $\hss}\vfil}
3149 \else
3150 \vbox to 4pt{\vss\hbox{ $\left#2\vrule width0pt height #3
3151 depth 0pt \right. $\}\vfil}
3152 \fi}

\edatright \edatright[ $]{\langle symbol \rangle}{\langle len \rangle}$  (combination and generalisation of original
\sekla and \sekla gl). Right  $\langle symbol \rangle$ ,  $2\langle len \rangle$  high with appended  $\langle math \rangle$  verti-
cally centered.

3153 \newcommand{\edatright}[3][\@empty]{%
3154 \ifx#1\@empty
3155 \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
3156 depth 0pt \right#2 $\hss}\vfil}
3157 \else
3158 \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
3159 depth 0pt \right#2 #1 $\}\vfil}
3160 \fi}
3161

\edvertline \edvertline{\langle len \rangle} vertical line  $\langle len \rangle$  high. (was \sestrich)
3162 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
3163

\edvertdots \edvertdots{\langle len \rangle} vertical dotted line  $\langle len \rangle$  high. (was \sepunkte)
3164 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
3165 {\cleaders\hbox{$\m@th\hbox{.}\vbox to 0.5em{ }\$}\vfil}}}
3166$ 
```

I don't know if this is relevant here, and I haven't tried it, but the following appeared on CTT.

From: mdw@nsict.org (Mark Wooding)
 Newsgroups: comp.text.tex
 Subject: Re: Dotted line
 Date: 13 Aug 2003 13:51:14 GMT

Alexis Eisenhofer <alexis@eisenhofer.de> wrote:

> Can anyone provide me with the LaTeX command for a vertical dotted line?

How dotted? Here's the basic rune.

```
\newbox\linedotbox
\setbox\linedotbox=\vbox{...}
\leaders\copy\linedotbox\vskip2in
```

For just dots, this works:

```
\setbox\linedotbox=\vbox{\hbox{\normalfont.}\kern2pt}
```

For dashes, something like

```
\setbox\linedotbox=\vbox{\leaders\vrule\vskip2pt\vskip2pt}
```

is what you want. (Adjust the '2pt' values to taste. The first one is the length of the dashes, the second is the length of the gaps.)

For dots in mid-paragraph, you need to say something like

```
\lower10pt\vbox{\leaders\copy\linedotbox\vskip2in}
```

which is scungy but works.

-- [mdw]

`\edfilldimen` A length. (was `\klamdimen`)

```
3167 \newdimen\edfilldimen
3168 \edfilldimen=0pt
3169
```

`\c@addcolcount` A counter to hold the number of a column. We use a roman number so that we can
`\theaddcolcount` grab the column dimension from `\dcol...`

```
3170 \newcounter{addcolcount}
3171 \renewcommand{\theaddcolcount}{\roman{addcolcount}}
```

`\l@dtabaddcols` `\l@dtabaddcols{<startcol>}{<endcol>}` adds the widths of the columns `<startcol>` through `<endcol>` to `\edfilldimen`. It is a LaTeX style reimplementation of the original `\@add@`.

```
3172 \newcommand{\l@dtabaddcols}[2]{%
3173   \l@dcheckstartend{#1}{#2}%
3174   \ifl@dstartendok
3175     \setcounter{addcolcount}{#1}%
3176     \@whilenum \value{addcolcount}<#2\relax \do
3177       {\advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
3178        \advance\edfilldimen by \edtabcolsep
3179        \stepcounter{addcolcount}}%
3180     \advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
3181     \fi
3182 }
3183
```

`\ifl@dstartendok` `\l@dcheckstartend{<startcol>}{<endcol>}` checks that the values of `<startcol>` and `\l@dcheckstartend`

$\langle endcol \rangle$ are sensible. If they are then `\ifl@dstartendok` is set TRUE, otherwise it is set FALSE.

```

3184 \newif\ifl@dstartendok
3185 \newcommand{\l@dcheckstartend}[2]{%
3186   \l@dstartendoktrue
3187   \ifnum #1<\@ne
3188     \l@dstartendokfalse
3189     \led@err@LowStartColumn
3190   \fi
3191   \ifnum #2>30\relax
3192     \l@dstartendokfalse
3193     \led@err@HighEndColumn
3194   \fi
3195   \ifnum #1>#2\relax
3196     \l@dstartendokfalse
3197     \led@err@ReverseColumns
3198   %% \ledmac@error{Start column is greater than end column}{\@ehc}%
3199   \fi
3200 }
3201
```

`\edrowfill` `\edrowfill{ $\langle startcol \rangle$ }{ $\langle endcol \rangle$ }` fill fills columns $\langle startcol \rangle$ to $\langle endcol \rangle$ inclusive with $\langle fill \rangle$ (e.g. `\hrulefill`, `\upbracefill`). This is a LaTeX style reimplementa-tion and generalization of the original `\waklam`, `\Waklam`, `\waklamec`, `\wastricht` and `\wapunktel` macros.

```

3202 \newcommand*\edrowfill}[3]{%
3203   \l@dtabaddcols{#1}{#2}%
3204   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
3205 \let\@edrowfill=\edrowfill
3206 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
3207
```

`\edbeforetab` The macro `\edbeforetab{ $\langle text \rangle$ }{ $\langle math \rangle$ }` puts $\langle text \rangle$ at the left margin before array cell entry $\langle math \rangle$. Conversely, the macro `\edaftertab{ $\langle math \rangle$ }{ $\langle text \rangle$ }` puts $\langle text \rangle$ at the right margin after array cell entry $\langle math \rangle$. `\edbeforetab` should be in the first column and `\edaftertab` in the last column. The following macros support these.

`\leftltab` `\leftltab{ $\langle text \rangle$ }` for `\edbeforetab` in `\ltab`. (was `\linksltab`)

```

3208 \newcommand{\leftltab}[1]{%
3209   \hb@xt@ \z@{\vbox{\edtabindent%
3210     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
3211
```

`\leftrtab` `\leftrtab{ $\langle text \rangle$ }{ $\langle math \rangle$ }` for `\edbeforetab` in `\rtab`. (was `\linksrtab`)

```

3212 \newcommand{\leftrtab}[2]{%
3213   #2\hb@xt@ \z@{\vbox{\edtabindent%
3214     \advance\Hilfsskip by\dcoli%
3215     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}

```

3216

`\leftctab` `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`. (was `\linksztab`)

```
3217 \newcommand{\leftctab}[2]{%
3218     \hb@xt@#2{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
3219     \advance\Hilfsskip by 0.5\dcoli%
3220     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3221     \disablel@dtabfeet$\displaystyle{#2}$}%
3222     \advance\Hilfsskip by -0.5\wd\hilfsbox%
3223     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
3224     #2}
3225 }
```

`\rightctab` `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`. (was `\rechtsztab`)

```
3226 \newcommand{\rightctab}[2]{%
3227     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3228     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
3229     #1\hb@xt@#2{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
3230     \advance\Hilfsskip by 0.5\l@dcolwidth%
3231     \advance\Hilfsskip by -\wd\hilfsbox%
3232     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3233     \disablel@dtabfeet$\displaystyle{#1}$}%
3234     \advance\Hilfsskip by -0.5\wd\hilfsbox%
3235     \advance\Hilfsskip by \edtabcolsep%
3236     \moveright\Hilfsskip\hbox{ #2}}\hss}%
3237     }
3238 }
```

`\rightltab` `\rightltab{<math>}{<text>}` for `\edaftertab` in `\ltab`. (was `\rechtsltab`)

```
3239 \newcommand{\rightltab}[2]{%
3240     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3241     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
3242     #1\hb@xt@#2{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
3243     \advance\Hilfsskip by\l@dcolwidth%
3244     \advance\Hilfsskip by-\wd\hilfsbox%
3245     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3246     \disablel@dtabfeet$\displaystyle{#1}$}%
3247     \advance\Hilfsskip by-\wd\hilfsbox%
3248     \advance\Hilfsskip by\edtabcolsep%
3249     \moveright\Hilfsskip\hbox{ #2}}\hss}%
3250     }
3251 }
```

`\rightrtab` `\rightrtab{<math>}{<text>}` for `\edaftertab` in `\rtab`. (was `\rechtsrtab`)

```
3252 \newcommand{\rightrtab}[2]{%
3253     \setbox\hilfsbox=\hbox{\def\edlabel##1{%
3254     \disablel@dtabfeet#2}%
3255     #1\hb@xt@#2{\vbox{\edtabindent%
3256     \advance\Hilfsskip by-\wd\hilfsbox%
```

```

3257      \advance\Hilfsskip by\edtabcolsep%
3258      \moveright\Hilfsskip\hbox{ #2}\hss}%
3259      }
3260

```

`\rtab` `\rtab{<body>}` typesets `<body>` as an array with the entries right justified. (was `\edbeforetab` `\rtab`) (Here and elsewhere, `\edbeforetab` and `\edaftertab` were originally `\davor` and `\danach`) The original `\rtab` and friends included a fair bit of common code which I have extracted into macros.

The process is first to measure the `<body>` to get the column widths, and then in a second pass to typeset the body.

```

3261 \newcommand{\rtab}[1]{%
3262   \l@dnnullfills
3263   \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
3264   \def\edaftertab##1##2{\righttab{##1}{##2}}%
3265   \measurebody{#1}%
3266   \l@drestorefills
3267   \variab
3268   \setmrowright #1\&\&\&%
3269   \enablel@dtabfeet}
3270

```

`\measurebody` `\measurebody{<body>}` measures the array `<body>`.

```

3271 \newcommand{\measurebody}[1]{%
3272   \disablel@dtabfeet%
3273   \l@dcolcount=0%
3274   \nullsetzen%
3275   \l@dcolcount=0
3276   \measuremrow #1\&\&\&%
3277   \global\l@dampcount=1}
3278

```

`\rtabtext` `\rtabtext{<body>}` typesets `<body>` as a tabular with the entries right justified. (was `\rtabtext`)

```

3279 \newcommand{\rtabtext}[1]{%
3280   \l@dnnullfills
3281   \measuretbody{#1}%
3282   \l@drestorefills
3283   \variab
3284   \setthrowright #1\&\&\&%
3285   \enablel@dtabfeet}
3286

```

`\measuretbody` `\measuretbody{<body>}` measures the tabular `<body>`.

```

3287 \newcommand{\measuretbody}[1]{%
3288   \disablel@dtabfeet%
3289   \l@dcolcount=0%
3290   \nullsetzen%
3291   \l@dcolcount=0

```

```

3292 \measuretrrow #1\\&\\%
3293 \global\l@dampcount=1}
3294

```

\ltab Array with entries left justified. (was \ltab)

```

\edbeforetab 3295 \newcommand{\ltab}[1]{%
\edaftertab 3296 \l@dnnullfills
3297 \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
3298 \def\edaftertab##1##2{\rightltab{##1}{##2}}%
3299 \measuretbody{#1}%
3300 \l@drestorefills
3301 \variab
3302 \setmrowleft #1\\&\\%
3303 \enablel@dtabfeet}
3304

```

\ltabtext Tabular with entries left justified. (was \ltabtext)

```

3305 \newcommand{\ltabtext}[1]{%
3306 \l@dnnullfills
3307 \measuretbody{#1}%
3308 \l@drestorefills
3309 \variab
3310 \settrrowleft #1\\&\\%
3311 \enablel@dtabfeet}
3312

```

\ctab Array with centered entries. (was \ztab)

```

\edbeforetab 3313 \newcommand{\ctab}[1]{%
\edaftertab 3314 \l@dnnullfills
3315 \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
3316 \def\edaftertab##1##2{\rightctab{##1}{##2}}%
3317 \measuretbody{#1}%
3318 \l@drestorefills
3319 \variab
3320 \setmrowcenter #1\\&\\%
3321 \enablel@dtabfeet}
3322

```

\ctabtext Tabular with entries centered. (new)

```

3323 \newcommand{\ctabtext}[1]{%
3324 \l@dnnullfills
3325 \measuretbody{#1}%
3326 \l@drestorefills
3327 \variab
3328 \settrrowcenter #1\\&\\%
3329 \enablel@dtabfeet}
3330

```

\spreadtext (was \breitertext)


```

3331 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
3332   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}

```

`\spreadmath` (was `\breiter`, ‘breiter’ = ‘broadly’)

```

3333 \newcommand{\spreadmath}[1]{%
3334   \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
3335

```

I have left the remaining TABMAC alone, apart from changing some names. I’m not yet sure what they do or how they do it. Authors should not use any of these as they are likely to be mutable.

`\tabellzwischen` (was `\tabellzwischen`)

```

3336 \def\tabellzwischen #1&{%
3337   \ifx #1\ \let\NEXT\relax \l@dcolcount=0
3338   \else \stepl@dcolcount%
3339         \l@dcolwidth = #1 mm
3340         \let\NEXT=\tabellzwischen
3341   \fi \NEXT }
3342

```

`\edatabell` For example `\edatabell 4 & 19 & 8 \` specifies 3 columns with widths of 4, 19, and 8mm. (was `\atabell`)

```

3343 \def\edatabell #1\{%
3344   \tabellzwischen #1&\&}

```

`\Setzen` (was `\Setzen`, ‘setzen’ = ‘set’)

```

3345 \def\Setzen #1&{%
3346   \ifx #1\relax \let\NEXT=\relax
3347   \else \stepl@dcolcount%
3348         \let\tabelskip=\l@dcolwidth
3349         \EDTAB #1|
3350         \let\NEXT=\Setzen
3351   \fi\NEXT}
3352

```

`\EDATAB` (was `\ATAB`)

```

3353 \def\EDATAB #1\{%
3354   \ifx #1\Relax \centerline{\Setzen #1\relax&}
3355         \let\Next\relax
3356   \else \centerline{\Setzen #1&\relax&}
3357         \let\Next=\EDATAB
3358   \fi\Next}

```

`\edatab` (was `\atab`)

```

3359 \newcommand{\edatab}[1]{%
3360   \variab%
3361   \EDATAB #1\\\Relax\\}
3362

```

```

\HILFSskip More helpers.
\Hilfsskip 3363 \newskip\HILFSskip
          3364 \newskip\Hilfsskip
          3365

\EDTABINDENT (was \TABINDENT)
          3366 \newcommand{\EDTABINDENT}{%
          3367   \ifnum\l@dc@lcount=30\let\NEXT\relax\l@dc@lcount=0%
          3368   \else\step1@dc@lcount%
          3369     \advance\Hilfsskip by\l@dc@lwidth%
          3370     \ifdim\l@dc@lwidth=0pt\advance\hilfsc@unt\@ne
          3371     \else\advance\Hilfsskip by \the\hilfsc@unt\edtabcolsep%
          3372     \hilfsc@unt=1\fi%
          3373     \let\NEXT=\EDTABINDENT%
          3374   \fi\NEXT}%

\edtabindent (was \tabindent)
          3375 \newcommand{\edtabindent}{%
          3376   \l@dc@lcount=0\relax
          3377   \Hilfsskip=0pt%
          3378   \hilfsc@unt=1\relax
          3379   \EDTABINDENT%
          3380   \hilfsskip=\hsize%
          3381   \advance\hilfsskip -\Hilfsskip%
          3382   \Hilfsskip=0.5\hilfsskip%
          3383   }%
          3384

\EDTAB (was \TAB)
          3385 \def\EDTAB #1|#2|{%
          3386   \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
          3387   \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
          3388   \advance\tabelskip -\wd\tabhilfbox%
          3389   \advance\tabelskip -\wd\tabHilfbox%
          3390   \unhbox\tabhilfbox\hskip\tabelskip%
          3391   \unhbox\tabHilfbox}%
          3392

\EDTABtext (was \TABtext)
          3393 \def\EDTABtext #1|#2|{%
          3394   \setbox\tabhilfbox=\hbox{#1}%
          3395   \setbox\tabHilfbox=\hbox{#2}%
          3396   \advance\tabelskip -\wd\tabhilfbox%
          3397   \advance\tabelskip -\wd\tabHilfbox%
          3398   \unhbox\tabhilfbox\hskip\tabelskip%
          3399   \unhbox\tabHilfbox}%

\tabhilfbox Further helpers.
\tabHilfbox 3400 \newbox\tabhilfbox

```

```
3401 \newbox\tabHilfbox
3402
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% That finishes tabmac
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
edarrayl The 'environment' forms for \ltab, \ctab and \rtab.
edarrayc 3403 \newenvironment{edarrayl}{\l@collect@body\ltab}{}
edarrayr 3404 \newenvironment{edarrayc}{\l@collect@body\ctab}{}
3405 \newenvironment{edarrayr}{\l@collect@body\rtab}{}
3406
```

```
edtabularl The 'environment' forms for \ltabtext, \ctabtext and \rtabtext.
edtabularc 3407 \newenvironment{edtabularl}{\l@collect@body\ltabtext}{}
edtabularr 3408 \newenvironment{edtabularc}{\l@collect@body\ctabtext}{}
3409 \newenvironment{edtabularr}{\l@collect@body\rtabtext}{}
3410
```

Here's the code for enabling \edtext (instead of \critext).

```
\usingcritext Declarations for using \critext{}.../ or using \edtext{}{} inside tabulars. The
\disablel@dtabfeet default at this point is for \edtext.
\enablel@dtabfeet 3411 \newcommand{\usingcritext}{%
\usingedtext 3412 \def\disablel@dtabfeet{\l@dmodforcritext}%
3413 \def\enablel@dtabfeet{\l@drestoreforcritext}}
3414 \newcommand{\usingedtext}{%
3415 \def\disablel@dtabfeet{\l@dmodforedtext}%
3416 \def\enablel@dtabfeet{\l@drestoreforedtext}}
3417
3418 \usingedtext
3419
```

33 The End

i/codei

A Examples

This section presents some sample documents.

The examples in sections A.2 through A.5, plus A.7, were originally written for TeX. I have done some limited conversions of these so that they look more like LaTeX code. In particular wherever possible I have replaced `\def` commands by either `\newcommand` or `\renewcommand` as appropriate. I have also replaced the original TeX font handling commands by the LaTeX font commands.

The other examples were written natively in LaTeX.

The figures are from processed versions of the files. Having latexed a file I used DVIPS to get Encapsulated PostScript, then the `epstopdf` script to get a PDF version as well, for example:

```
> latex ledeasy
> latex ledeasy
> latex ledeasy
> dvips -E -o ledeasy.eps ledeasy
> epstopdf ledeasy.eps    % produces ledeasy.pdf
```

For those who aren't fascinated by LaTeX code, I show the all the typeset results first, then the code that produced them.

Simple Example

Peter Wilson*

Contents

1	First	1
1.1	Example text	1
2	Last	1

1 First

This is a simple example of using the `ledmac` package with ordinary LaTeX constructs.

1.1 Example text

- 1 The `ledmac` package lets you do some unusual things in a LaTeX document.
- 2 For example you can have lines numbered and there are several levels of foot-
- 3 notes. You can label lines within the numbered text and refer to them outside. Sidenotes
- 4 Do not try and use any normal LaTeX `marginpars`¹ or `exotica` within the num- are OK
- 5 bered portions of the text.

2 Last

I forgot to mention that you can use ordinary footnotes^{2,3} outside the numbered text. You can also^a have^b formatted footnotes^c in normal^d text.

There are 5 numbered lines in the example shown in section 1.1.

*Standing on the shoulders of giants.

¹You will get a warning but no text.

²An ordinary footnote

³And another

^aAdditionally ^bSpecify ^cLike this ^dText that does not have line numbers

2 several] This is an 'A' footnote.

4 `exotica`] Like floats.

2 levels] This is a 'B' level footnote.

Figure 1: Output from `ledeasy.tex`.

This is an example of some text with variant readings recorded as ‘A’ foot-
notes. From here on, though, we shall have ‘C’. For spice, let us mark a longer
3 passage, but give a different lemma for it, so that we don’t get a huge amount
4 of text in a note. Finally, we shouldn’t forget the paragraphed notes, which are
5 so useful when there are a great number of short notes to be recorded.
6 This is a second paragraph, giving more *examples* of text with variant read-
7 ings recorded as ‘A’ footnotes. From here on, though, we shall have ‘B’ notes in
8 the text. For spice, let us mark a longer passage, but give a different lemma for
9 it, so that we don’t get a *huge* amount of text in a note. Finally, we shouldn’t
10 forget the column notes, which are so useful when there are many short notes
11 to be recorded.

1 example:: eximemple C, D.	6 examples:: eximples L, M.
1 variant:: alternative, A, B.	6 variant:: alternative, A, B.
2 though:: however α, β	
2 ‘C’] B, <i>pace</i> the text	9 shouldn’t] ought not to
7 though] however α, β	L, M
7 ‘B’] B, as correctly	10 forget the] omit to
stated in the text	mention the §, ¶
9 Finally] In the end X,	10 column] blocked M, N
Y	10 notes] variants H
9 we] we here K	10 useful] very, very useful
	L, P
	10 many] lots of Z
	11 recorded] recorded and
	put down: M
	(repetition)
2–4 For spice ... note: The note here is type ‘C’	
8–9 For spice, ... note: This is a rogue note of type ‘C’.	
3 huge: vast E, F; note that this is a ‘D’ note to section of text within a longer lemma	
9 huge: vast E, F; note that this is a ‘D’ note to text within a longer lemma.	
4 Finally: in the end X, Y	4 we: us K
4 shouldn’t: ought not to L, M	4 forget
the: omit to mention the §, ¶	4 paragraphed: blocked M, N
4 notes: variants HH, KK	
5 useful: truly useful L, P	5 a great number of: many, many (preferably)
5 recorded:	5 recorded:
noted: repetition	

Figure 2: Output from `ledfeat.tex`.

Oedipus entreth.

Or that with wrong the right and doubtlesse heire,
 Shoulde banisht be out of his princely seate.
 Yet thou O queene, so fyle thy sugred tounge,
 And with suche counsell decke thy mothers tale,
 That peace may bothe the brothers heartes inflame, 5
 And rancour yelde, that erst possesse the same.

Eteocl. Mother, beholde, youre hestes for to obey,
 In person nowe am I resorted hither:
 In haste therefore, fayne woulde I knowe what cause
 With hastie speede, so moued hath your mynde 10
 To call me nowe so causelesse out of tyme,
 When common wealth moste craues my onely ayde:
 Fayne woulde I knowe, what queynt commoditie
 Persuades you thus to take a truce for tyme,
 And yelde the gates wide open to my foe, 15
 The gates that myght our stately state defende,
 And nowe are made the path of our decay.

„ *Ioca.* Represse deare son, those raging stormes of wrath,
 „That so bedimme the eyes of thine intente,
 „As when the tongue (a redy Instrument) 20
 „Would fayne pronounce the meaning of the minde,
 „It cannot speake one honest seemely worde.
 „But when disdayne is shrunke, or sette asyde,
 „And mynde of man with leysure can discourse
 „What seemely woordes his tale may best beseeme, 25
 „And that the tounge vnfoldes without affectes
 „Then may proceede an answere sage and graue,
 „And euery sentence sawst with sobernesse:
 Wherefore vnbende thyne angrie browes deare chylde,
 And caste thy rolling eyes none other waye, 30
 That here doost not *Medusaes* face beholde,
 But him, euen him, thy blood and brother deare.
 And thou beholde, my *Polinices* eke,
 Thy brothers face, wherein when thou mayst see
 Thine owne image, remember therwithall, 35
 That what offence thou woldst to him were done,

0.1 entreth] *intrat* MS 20–22 As ... worde.] *not in* 73 20 the] *thie* MS 21 fayne
 pronounce] *faynest tell* MS 21 the minde] *thy minde* MS 22 It ... worde.] *Thie swelling*
 hart puft vp with wicked ire / Can scarce pronounce one inward louing thought. MS 31
Medusaes] One of the furies. 75m

[SCENE III.—*Venice*.]*Enter JESSICA and [LAUNCELOT] the clown.*

- Jes.* I am sorry thou wilt leave my father so,
 Our house is hell, and thou (a merry devil)
 Didst rob it of some taste of tediousness,—
 But fare thee well, there is a ducat for thee,
 And Launcelot, soon at supper shalt thou see 5
 Lorenzo, who is thy new master's guest,
 Give him this letter,—do it secretly,—
 And so farewell: I would not have my father
 See me in talk with thee.
- Laun.* Adieu! tears exhibit my tongue, most beautiful pagan, most sweet 10
 Jew!—if a Christian do not play the knave and get thee, I am much
 deceived; but adieu! these foolish drops do something drown my
 manly spirit: adieu! *[Exit.]*
- Jes.* Farewell good Launcelot.
 Alack, what heinous sin is it in me 15
 To be ashamed to be my father's child!

Scene III] *Capell*; *om.* *Q*, *F*; Scene IV *Pope*. *Venice*] *om.* *Q*, *F*; *Shylock's house* *Theobald*;
The same. A Room in Shylock's House *Capell*. Launcelot] *Rowe*; *om.* *Q*, *F*. 1. I am] *Q*,
F; I'm *Pope*. 9. in] *Q*; *om.* *F*. 10. *Laun.*] *Q2*; *Clowne*. *Q*, *F*. 10. Adieu!] Adieu, *Q*, *F*.
11. Jew!] Iewe, *Q*, *F*. do] *Q*, *F*; did *F2*. 12. adieu!] adieu, *Q*, *F*. 12. something] *Q*;
somewhat *F*. 13. adieu!] adieu. *Q*, *F*. S. D.] *Q2*, *F*; *om.* *Q*; after l. 15 *Capell*. 16. child!] *Q*, *F*;
Child? *Rowe*.

5. *soon*] early.

10. *exhibit*] *Eccles* paraphrased "My tears
 serve to express what my tongue should, if
 sorrow would permit it," but probably it is
 Launcelot's blunder for prohibit (*Halliwell*)
 or inhibit (*Clarendon*).

10. *pagan*] This may have a scurrilous un-
 dertone: cf. 2 *H* 4, II. ii. 168.

11. *do*] *Malone* upheld the reading of *Qq*

and *F* by comparing II. vi. 23: "When you
 shall please to play the thieves for wives";
 Launcelot seems fond of hinting at what is
 going to happen (cf. II. v. 22–3). If *F2*'s "did"
 is accepted, *get* is used for beget, as in III. v.
 9.

12–13. *foolish... spirit*] "tears do not be-
 come a man" (*AYL.*, III. iv. 3); cf. also *H* 5,
 IV. vi. 28–32.

Figure 4: Output from `ledarden.tex`.

1 incipit . . . ΠΕΡΙΦΥΣΕΩΝ] *om. R, incipit quartus M* 2 ΑΝΑΚΕΦΑΛΙΟΣΙΣ] *FJP, lege ἀνακεφαλαιώσεις* 2 physiologiae] *physiologiae P, physeologiae R* 3 quod] *p. natura transp.* *MR* 3 ΥΠΕΡΟΥΣΙΑΔΕΣ] *codd. Vtrum ὑπερουσιώδης (hoc est supersessentialis) natura cum Gale (p.160) an ὑπερουσιότης (hoc est supersessentialis natura) cum Floss (PL 122,741C) intelligendum sit, ambigitur* 7 ΟΜΟΥΣΙΟΣ] *codd., lege ὁμοούσιος* 7 *et*] *R¹, om. R⁰* 9 ΑΓΚΥΡΑΤΩ] *anchurato MR* 9 *de fide*] *Glo(ssa): Ita enim uocatur sermo eius de fide* ΑΓΚΥΡΑΤΟΣ, *id est procuratus mg. add. FJP* 10 *agentia*] *actiua MR* 10 *formantia*] *formatiua MR* 11 *operantia*] *operatiua MR* 13 *eiusdem*] *eiusdemque M* 13 *eiusdem* *uirtutis, eiusdem subsistentiae*] *om. M* 13 *subsistentiae*] *substantia R* 14 *similiter*] *ex simili MR* 15 *sunt*] *om. M* 25 *spiritus sanctus*] *sanctus spiritus R*

Chronicle of Guelders

Guillelmus de Berchen

St. Stephen's Church in Nijmegen

Nobilis itaque comes Otto imperio et dominio Novimagensi sibi, ut praefer-
tur, impignoratis et commissis proinde praeesse cupiens, anno LIII superius 1254
descripto, mense Iunio, una cum iudice, scabinis ceterisque civibus civitatis
Novimagensis, pro ipsius et inhabitantium in ea necessitate, commodo et utili-
tate, ut ecclesia eius parochialis extra civitatem sita destrueretur et infra muros
5 transferretur ac de novo construeretur, a reverendo patre domino Conrado de
Hofsteden, archiepiscopo Coloniensi, licentiam, et a venerabilibus dominis de-
cano et capitulo sanctorum Apostolorum Coloniensi, ipsius ecclesiae ab antiquo
veris et pacificis patronis, consensum, citra tamen praeiudicium, damnum aut
10 gravamen iurium et bonorum eorundem, impetravit.

Et exinde liberum locum eiusdem civitatis qui dicitur Hundisbrug, de prae-
libati Wilhelmi Romanorum regis, ipsius fundi domini, consensu, ad aedifican-
dum et consecrandum ecclesiam et coemeterium, eisdem decano et capitulo de
expresso eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
15 se ipsi comes et civitas dictis decano et capitulo, quod in recompensationem
illius areae infra castrum et portam, quae fuit dos ecclesiae, in qua plebanus
habitare solebat—quae tunc per novum fossatum civitatis est destructa—aliam
aream competentem et ecclesiae novae, ut praefertur, aedificandae satis conti-
guam, ipsi plebano darent et assignarent. Et desuper apud dictam ecclesiam
20 sanctorum Apostolorum est littera sigillis ipsorum Ottonis comitis et civitatis
Novimagensis sigillata.

3 p. 227 R 4 p. 97 N 6 p. 129 D 12 f. 72v M 13 p. 228 R 20 p. 130 D

2 proinde] primum D 5 ecclesia eius] ecclesia D: eius eius H extra civitatem *om.* H
infra] intra D 6 transferretur] transferreretur NH 7 Hofsteden] Hoffstede D: Hoffsteden
H Coloniensi] Colononiensi H dominis] viris H 8 Coloniensi] Coloniae H 10 iurium]
virium D 11 liberum] librum H qui] quae D Hundisbrug] Hundisburch D: Hunsdisbrug
R 12 regis] imperatoris D 13 et consecrandum *om.* H eisdem] eiusdem D 15 comes]
comites D dictis *om.* H 17 tunc] nunc H 18 ut...aedificandae *om.* H 18–19 contiguam]
contiguum M 19 apud *om.* H 20 est] et H littera] litteram H 21 Novimagensis]
Novimagii D sigillata] sigillis communita H

6–7 William is confusing two charters that are five years apart. Permission from St. Apost-
les' Church in Cologne had been obtained as early as 1249. Cf. Sloet, *Oorkondenboek* nr.
707 (14 November 1249): "...nos devotionis tue precibus annuentes, ut ipsam ecclesiam
faciens demoliri transferas in locum alium competentem, tibi auctoritate presentium indulge-
mus..." 6–7 Conrad of Hochstaden was archbishop of Cologne in 1238–1261 11–21 Cf.
Sloet, *Oorkondenboek* nr. 762 (June 1254)

Figure 6: Output from `ledecker.tex`.

22

[Seán Ó Braonáin cct] chuim Tomás Uí Dhúnlain
 [Fonn: Máirseáil Uí Shúilleabháin (Páinseach na nUbh)]

- 1** A dhuine gan chéill do mhaisligh an chlér
 b is tharcainnigh naomhscriupt na bhfáige,
 c na haitheanta réab 's an t-aifreann thréig
 d re taitheamh do chlaonchreideamh Mhártain,
 e cá rachair 'od dhíon ar Íosa Nasardha
 f nuair chaithfimid cruinn bheith ar mhaoileann
 Josepha?
 g Ní caraid Mac Crae chuim t'anama ' phlé
 h ná Calvin bhais taobh ris an lá sin.
- 2** Nách damanta an scéal don chreachaire chlaon
 b ghlac baiste na cléire 'na pháiste
 c 's do glanadh mar ghréin ón bpeaca ró-dhaor
 d trí ainibhfios Éva rinn Ádam,
 e tuitim arís fé chuing na haicme sin
 f tug atharrach brí don scríbhinn bheannaithe,
 g d'aistrigh béasa agus reachta na cléire
 h 's nách tugann aon ghéilleadh don Phápa?
- 3** Gach scolaire baoth, ní mholaim a cheird
 b 'tá ag obair le géilleadh dá tháille
 c don doirbhchoin chlaon dá ngorthar Mac Crae,
 d deisceabal straeigh as an gcolláiste.
 e Tá adaithe thíos in íochtar ifrinn,
 f gan solas gan soilse i dtíorthaibh dorchá,
 g tuigsint an léinn, gach cuirpeacht déin
 h is Lucifer aosta 'na mháistir.

22 Teideal: Dhuinnluinnng T, Seághan Mac Domhnaill cct B
 1.a dhuinne T 1.a mhaslaidh T, mhaslaig B 1.c raob T 1.d le B 1.e
 dod B 1.f chaithfamíd T 1.f maolinn B 1.g phleidl T 1.h bhíos B
 1.h leis B 2.a claon B 2.c glannuig T 2.d ainibhfios T, ainibhfios B
 2.d Éabha B 2.g is B 2.h tuigíonn T 3.a sgollaire T 3.a mholluim T
 3.b 'tág ccobar T 3.b re B 3.c dorbhchon daor B 3.d straodhaig T
 3.e fhadoghthe tsíos T 3.e fadaighthe B 3.f sollus T 3.g cuirripeacht T
 3.h Luicifer T, Lúcifer B 3.h mhaighistir T

Figure 7: Output from `ledbraonain.tex`.

A.1 Simple example

This made-up example, `ledeasy.tex`, is included to show how simple it can be to use EDMAC in a LaTeX document. The code is given below and the result is shown in Figure 1.

```

3420 (*easy)
3421 % ledeasy.tex simple example of the ledmac package
3422 \documentclass{article}
3423 \usepackage{ledmac}
3424 %% number every line
3425 \setcounter{firstlinenum}{1}
3426 \setcounter{linenumincrement}{1}
3427 %% Show some B series familiar footnotes, lettered and paragraphed
3428 \renewcommand*{\thefootnoteB}{\alph{footnoteB}}
3429 \footparagraphX{B}
3430 %% no endnotes
3431 \noendnotes
3432 %% narrow sidenotes
3433 \setlength{\ledrsnotewidth}{4em}
3434 \title{Simple Example}
3435 \author{Peter Wilson\thanks{Standing on the shoulders of giants.}}
3436 \date{}
3437 \begin{document}
3438 \maketitle
3439 \tableofcontents
3440 \section{First}
3441   This is a simple example of using the \textsf{ledmac}
3442 package with ordinary LaTeX constructs.
3443
3444 \subsection{Example text}\label{subsec}
3445
3446 \beginnumbering
3447 \pstart
3448 The \textsf{ledmac} package lets you do some unusual things in
3449 a LaTeX document. For example you can have lines numbered and
3450 there are
3451 \edtext{several}{\Afootnote{This is an ‘A’ footnote.}}
3452 \edtext{levels}{\Bfootnote{This is a ‘B’ level footnote.}}
3453 of footnotes.
3454 You can label lines within the numbered text and refer to them
3455 outside. Do not try and use any normal LaTeX
3456 \marginpars\footnote{You will get a warning but no text.}%
3457 \ledrightrightnote{Sidenotes are OK}
3458 or \edtext{exotica}{\Afootnote{Like floats.}}
3459 within the numbered portions of the text\edlabel{line}.
3460 \pend
3461 \endnumbering
3462
3463 \section{Last}

```

```

3464
3465     I forgot to mention that you can use ordinary
3466 \footnote{An ordinary footnote}\footnote{And another}
3467 outside the numbered text. You can also\footnoteB{Additionally}
3468 have\footnoteB{Specify} formatted footnotes\footnoteB{Like this}
3469 in normal\footnoteB{Text that does not have line numbers} text.
3470
3471     There are \lineref{line} numbered lines in the example shown
3472 in section~\ref{subsec}.
3473
3474 \end{document}
3475 </easy>

```

A.2 General example of features

This made-up example, `ledfeat.tex`, is included purely to illustrate some of `ledmac`'s main features. It is hard to find real-world examples that actually use as many layers of notes as this, so we made one up. The example is a bit tricky to read, but close study and comparison with the output (Figure 2) will be illuminating.

I have converted the original TeX code to look more like LaTeX code.

```

3476 <*features>
3477 % ledfeat.tex  Small test file for ledmac package
3478 \documentclass{article}
3479 \usepackage{ledmac}
3480
3481 \noendnotes % we aren't having any endnotes
3482
3483 \makeatletter
3484 % I'd like a spaced out colon after the lemma:
3485 \newcommand{\spacedcolon}{\rmfamily\thinspace:\thinspace}
3486 \renewcommand*{\normalfootfmt}[3]{%
3487   \ledsetnormalparstuff
3488   {\notenumfont\printlines#1|}\strut\enspace
3489   {\select@lemmafont#1/#2}\spacedcolon\enskip#3\strut\par}
3490
3491 % And I'd like the 3-col notes printed with a hanging indent:
3492 \renewcommand*{\threecolfootfmt}[3]{%
3493   \normal@pars
3494   \hsize .3\hsize
3495   \setlength{\parindent}{0pt}
3496   \tolerance=5000 % high, but not infinite
3497   \raggedright
3498   \hangindent1.5em \hangafter1
3499   \leavevmode
3500   \strut\hbox to 1.5em{\notenumfont\printlines#1|\hfil}\ignorespaces
3501   {\select@lemmafont#1/#2}\rbracket\enskip
3502   #3\strut\par\allowbreak}
3503

```

```

3504 % And I'd like the 2-col notes printed with a double colon:
3505 \newcommand*{\doublecolon}{\rmfamily\thinspace:\thinspace}}
3506 \renewcommand*{\twocolfootfmt}[3]{%
3507   \normal@pars
3508   \hsize .45\hsize
3509   \setlength{\parindent}{0pt}
3510   \tolerance=5000
3511   \raggedright
3512   \leavevmode
3513   \strut{\notenumfont\printlines#1/}\enspace
3514   {\select@lemmafnt#1/#2}\doublecolon\enskip
3515   #3\strut\par\allowbreak}
3516
3517 % And in the paragraphed footnotes, I'd like a colon too:
3518 \renewcommand*{\parafootfmt}[3]{%
3519   \ledsetnormalparstuff
3520   {\notenumfont\printlines#1/}\enspace
3521   {\select@lemmafnt#1/#2}\spacedcolon\enskip
3522   #3\penalty-10 }
3523 \makeatother
3524
3525 % I'd like the line numbers picked out in bold.
3526 \renewcommand{\notenumfont}{\bfseries}
3527 \lineation{page}
3528 \linenummargin{inner}
3529 \setcounter{firstlinenum}{3}          % just because I can
3530 \setcounter{linenumincrement}{1}
3531 \foottwocol{A}
3532 \footthreecol{B}
3533 \footparagraph{E}
3534 % I've changed \normalfootfmt, so invoke it again for C and D notes.
3535 \footnormal{C}
3536 \footnormal{D}
3537
3538 \begin{document}
3539
3540 \begin{numbering}
3541
3542 \pstart
3543 This is an \edtext{example}{
3544   \Afootnote{eximemple C, D.}}
3545 of some %\footnote{A normal footnote}
3546 text with \edtext{variant}{
3547   \Afootnote{alternative, A, B.}}
3548 readings recorded as 'A' footnotes. From here on, \edtext{though}{
3549   \Afootnote{however $\alpha$, $\beta$}},
3550 we shall have \edtext{'C'}{
3551   \Bfootnote{B, \textit{pace} the text}}.
3552 \edtext{For spice, let us mark a longer passage, but give a different
3553   lemma for it, so that we don't get a \edtext{huge}{

```

```

3554     \Dfootnote{vast E, F; note that this is
3555         a 'D' note to section of text within a longer lemma}}
3556     amount of text in a note}{\lemma{For spice \dots\ note}
3557     \Cfootnote{The note here is type 'C'}}.
3558 \edtext{Finally}{
3559     \Efootnote{in the end X, Y}},
3560 \edtext{we}{
3561     \Efootnote{us K}}
3562 \edtext{shouldn't}{
3563     \Efootnote{ought not to L, M}}
3564 \edtext{forget the}{
3565     \Efootnote{omit to mention the \S, \P}}
3566 \edtext{paragraphed}{
3567     \Efootnote{blocked M, N}}
3568 \edtext{notes}{
3569     \Efootnote{variants HH, KK}},
3570 which are so \edtext{useful}{
3571     \Efootnote{truly useful L, P}}
3572 when there are \edtext{a great number of}{
3573     \Efootnote{many, many (preferably)}}
3574 short notes to be \edtext{recorded}{
3575     \Efootnote{noted: repetition}}.
3576 \pend
3577
3578 \pstart
3579 This is a second paragraph, giving more \textit{\edtext{examples}}{
3580     \Afootnote{eximples L, M.}}}
3581 of text with \edtext{variant}{
3582     \Afootnote{alternative, A, B.}}
3583 readings recorded as 'A' footnotes. From here on, \edtext{though}{
3584     \Bfootnote{however $\alpha$, $\beta$}},
3585 we shall have \edtext{'B'}{
3586     \Bfootnote{B, as correctly stated in the text}} notes in the text.
3587 \edtext{For spice, let us mark a longer passage, but give a different
3588 lemma for it, so that we don't get a \textit{\edtext{huge}}{
3589     \Dfootnote{vast E, F; note that this is
3590         a 'D' note to text within a longer lemma.}}}
3591 amount of text in a note}{\lemma{For spice, \dots\ note}
3592     \Cfootnote{This is a rogue note of type 'C'.}}.
3593 \edtext{Finally}{
3594     \Bfootnote{In the end X, Y}},
3595 \edtext{we}{
3596     \Bfootnote{we here K}}
3597 \edtext{shouldn't}{
3598     \Bfootnote{ought not to L, M}}
3599 \edtext{forget the}{
3600     \Bfootnote{omit to mention the \S, \P}}
3601 \edtext{column}{
3602     \Bfootnote{blocked M, N}}
3603 \edtext{notes}{

```

```

3604 \Bfootnote{variants H}},
3605 which are so \edtext{useful}{
3606 \Bfootnote{very, very useful L, P}}
3607 when there are \edtext{many}{
3608 \Bfootnote{lots of Z}}
3609 short notes to be \edtext{recorded}{
3610 \Bfootnote{recorded and put down: M (repetition)}}.
3611 \pend
3612
3613 \endnumbering
3614 \end{document}
3615 </features>

```

A.3 Gascoigne

The first real-life example is taken from an edition of George Gascoigne's *A Hundreth Sundrie Flowres* that is being prepared by G. W. Pigman III, at the California Institute of Technology. Figure 3 shows the result of setting the text with `ledmac`.

I have LaTeXified the original code, and removed all the code related to the main document layout, relying on the standard LaTeX layout parameters..

```

3616 (*ioc)
3617 %% ledioc.tex
3618 \documentclass{article}
3619 \usepackage{ledmac}
3620
3621 \noendnotes
3622 \makeatletter
3623
3624 \newcommand{\os}{\scriptsize}
3625 \setcounter{firstsublinenum}{1000}
3626 \frenchspacing \setlength{\parskip}{0pt} \hyphenpenalty=1000
3627
3628 % Say \nolinenums if you want no line numbers in the notes.
3629 \newif\ifnolinenums
3630 \newcommand{\nolinenums}{\global\nolinenumstrue}
3631 \newcommand{\linenums}{\global\nolinenumsfalse}
3632
3633 \renewcommand{\rightlinenum}{\ifbypage@\ifnum\line@num<10\kern.5em\fi\else
3634 \ifnum\line@num<10\kern1em\else\ifnum\line@num<100
3635 \kern.5em\fi\fi\fi\kern.5em\numlabfont\the\line@num
3636 \ifnum\subline@num>0:\the\subline@num\fi}
3637
3638 \renewcommand{\leftlinenum}{\numlabfont\the\line@num
3639 \ifnum\subline@num>0:\the\subline@num\fi \kern.5em}
3640 \linenummargin{outer}
3641 \lineation{page}

```



```

3642
3643 \newcommand{\ggfootfmt}[3]{%
3644   \notefontsetup
3645   \let\par=\endgraf
3646   \rightskip=0pt \leftskip=0pt
3647   \setlength{\parindent}{0pt} \parfillskip=0pt plus 1fil
3648   \ifnolinenums\relax\else
3649     \begingroup \os \printlines#1\endgroup
3650     \enskip
3651   \fi
3652   {\rmfamily #2\def\@tempa{#2}\ifx\@tempa\empty
3653     \else\enskip\fi#3\penalty-10 }}
3654
3655 % Now reset the \Afootnote parameters and macros:
3656 \footparagraph{A}
3657 \let\Afootfmt=\ggfootfmt
3658 \dimen\Afootins=\vsize
3659 \skip\Afootins=3pt plus9pt
3660 \newcommand*{\ggfootstart}[1]{\vskip\skip\Afootins}
3661 \let\Afootstart=\ggfootstart
3662
3663 \newcommand*{\stage}[1]{\pstart\startsub\parindent=0pt
3664   \hangindent=3em\hangafter=0
3665   {\itshape #1}\let\par=\finishstage}
3666 \newcommand{\finishstage}{\pend\endsub}
3667 \newcommand{\sen}{\leavevmode\lowerlex\hbox{\textrm{'}}}%
3668 \newcommand{\senspeak}[1]{\pstart\obeylines\setbox0=\hbox{\textrm{'}}}%
3669   \leavevmode
3670   \lowerlex\copy0\kern-\wd0\hskip1em{\textit{#1}}}%
3671   \hbox tolex{\ignorespaces}
3672 \newcommand*{\speak}[1]{\pstart\obeylines\hskip1em{\textit{#1}}}%
3673   \hbox tolex{\ignorespaces}
3674 \def\nospeaker{\parindent=0em\pstart\let\par=\pend}
3675 \newcommand*{\nospeak}{\pstart\obeylines}
3676 \makeatother
3677
3678 \begin{document}
3679
3680 \setlength{\parindent}{0pt}
3681
3682 \beginnumbering
3683
3684 \stage{Oedipus \edtext{entreth}{\Afootnote{\textit{intrat} MS}}.}
3685
3686 \nospeak
3687 Or that with wrong the right and doubtlesse heire,
3688 Shoulde banisht be out of his princely seate.
3689 Yet thou O queene, so fyle thy sugred tounge,
3690 And with suche counsell decke thy mothers tale,
3691 That peace may bothe the brothers heartes inflame,

```

```

3692 And rancour yelde, that erst possesse the same.
3693 \pend
3694
3695 \speak{Eteocl.} Mother, beholde, youre hestes for to obey,
3696 In person nowe am I resorted hither:
3697 In haste therefore, fayne woulde I knowe what cause
3698 With hastie speede, so moued hath your mynde
3699 To call me nowe so causelesse out of tyme,
3700 When common wealth moste craues my onely ayde:
3701 Fayne woulde I knowe, what queynt commoditie
3702 Persuades you thus to take a truce for tyme,
3703 And yelde the gates wide open to my foe,
3704 The gates that myght our statly state defende,
3705 And nowe are made the path of our decay.
3706 \pend
3707
3708 \sensspeak{Ioca.} Represse deare son, those raging stormes of wrath,
3709 \sen That so bedimme the eyes of thine intende,
3710 \edtext{\sen As when \edtext{the}{\Afootnote{thie MS}} tongue %
3711 (a redy Instrument)
3712 \sen Would \edtext{fayne pronounce}{\Afootnote{faynest tell MS}} %
3713 the meaning of \edtext{the minde}{\Afootnote{thy minde MS}},
3714 \sen \edtext{It}{\lemma{It \dots\ worde.}\Afootnote{Thie %
3715 swelling hart puft vp with wicked ire / Can scarce pronounce %
3716 one inward louing thought. MS}} cannot speake one honest %
3717 seemely worde.}{\lemma{As \dots\ worde.}\Afootnote{\textit{not %
3718 in} \os73}}
3719 \sen But when disdayne is shrunke, or sette asyde,
3720 \sen And mynde of man with leysure can discourse
3721 \sen What seemely wordes his tale may best beseeme,
3722 \sen And that the tounge vnfoldes without affectes
3723 \sen Then may proceede an answere sage and graue,
3724 \sen And euery sentence sawst with sobernesse:
3725 Wherefore vnbende thyne angrie browes deare chylde,
3726 And caste thy rolling eyes none other waye,
3727 That here doost not \edtext{\textit{Medusaes}}{%
3728 \Afootnote{One of the furies. {\os75m}} face beholde,
3729 But him, euen him, thy blood and brother deare.
3730 And thou beholde, my \textit{Polinices} eke,
3731 Thy brothers face, wherein when thou mayst see
3732 Thine owne image, remember therewithall,
3733 That what offence thou woldst to him were done,
3734 \pend
3735 \endnumbering
3736
3737 \end{document}
3738
3739 \end{document}

```

A.4 Shakespeare

The following text illustrates another input file of moderate complexity, with two layers of annotation in use. The example is taken from the Arden *Merchant of Venice*.

I have roughly converted the original TeX file to a LaTeX file. The file is below and the result of LaTeXing it is shown in Figure 4.

```

3740 (*arden)
3741 %% ledarden.tex
3742 \documentclass{article}
3743 \usepackage{ledmac}
3744
3745 \makeatletter
3746 \newcommand{\stage}[1]{\rlap{\hbox to \the\linenumsep{%
3747     \hfil\llap{[\textit{#1}]}}}
3748
3749 \newcommand{\speaker}[1]{\pstart\hangindent2em\hangafter1
3750     \leavevmode\textit{#1}\enspace\ignorespaces}
3751
3752 \newcommand{\exit}[1]{\hfill\stage{#1}}
3753
3754 % LEDMAC customizations:
3755 \noendnotes
3756 \setlength{\parindent}{0pt}
3757 \setlength{\linenumsep}{.4in}
3758 \rightskip\linenumsep
3759
3760 \renewcommand{\interparanoteglue}{1em plus.5em minus.1em}
3761
3762 \newcommand{\scf}{\tiny}
3763 \let\Afootnoterule=\relax \let\Bfootnoterule=\relax
3764
3765 \renewcommand{\rightlinenum}{\numlabfont\llap{\the\line@num}}
3766 \frenchspacing
3767
3768 % Footnote formats:
3769 % \nonumparafootfmt is a footnote format without line numbers.
3770 \newcommand{\nonumparafootfmt}[3]{%
3771     \ledsetnormalparstuff
3772     \rightskip=0pt
3773     \select@lemmafnt#1/#2\rbracket\enskip
3774     \itshape #3\penalty-10 }
3775
3776 \newcommand{\newparafootfmt}[3]{%
3777     \ledsetnormalparstuff
3778     {\notenumfont\printlines#1/}\fullstop\enspace
3779     {\select@lemmafnt#1/#2\rbracket\enskip
3780     \itshape #3\penalty-10 }
3781

```

```

3782 \newcommand{\newtwocolfootfmt}[3]{%
3783   \normal@pars
3784   \hsize .48\hsize
3785   \tolerance=5000
3786   \rightskip=0pt \leftskip=0pt \parindent=5pt
3787   \strut\notenumfont\printlines#1/\fullstop\enspace
3788   \itshape #2/>\rbracket\penalty100\hskip .5em plus .5em
3789   \normalfont #3\strut\goodbreak}
3790
3791 % Footnote style selections etc. (done last):
3792 \footparagraph{A}
3793 \foottwocol{B}
3794 \let\Afootfmt=\newparafootfmt
3795 \let\Bfootfmt=\newtwocolfootfmt
3796 \let\collation=\Afootnote
3797 \let\note=\Bfootnote
3798 \lineation{section}
3799 \linenummargin{right}
3800 \makeatother
3801
3802 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
3803
3804 \begin{document}
3805   \pagestyle{empty}
3806
3807   % Initially, we don't want line numbers.
3808   \let\Afootfmt=\nonumparafootfmt
3809
3810   \beginnumbering
3811   \pstart
3812   \centerline{[\edtext{SCENE III}{
3813     \lemma{Scene III}
3814     \collation{Capell; om. Q, F; \textnormal{Scene IV} Pope.}}.---%
3815     \edtext{\textit{Venice}}}{
3816     \collation{om. Q, F; Shylock's house Theobald; The same.
3817     A Room in Shylock's House Capell.}}.]}
3818   \pend
3819   \bigskip
3820
3821   \pstart
3822   \centerline{\textit{Enter} JESSICA \textit{and}
3823     [\edtext{LAUNCELOT}{
3824       \lemma{Launcelot}
3825       \collation{Rowe; om. Q, F.}}] \textit{the clown.}} \pend \bigskip
3826
3827   \let\Afootfmt=\newparafootfmt % we do want line numbers from now
3828
3829   \setline{0}%
3830
3831   \speaker{Jes.}\edtext{I am}{

```

```

3832 \collation{Q, F; \textnormal{I'm} Pope.}}
3833         sorry thou wilt leave my father so,\
3834 Our house is hell, and thou (a merry devil)\
3835 Didst rob it of some taste of tediousness,---\
3836 But fare thee well, there is a ducat for thee,\
3837 And Launcelot, \edtext{soon}{
3838     \note{early.}}
3839         at supper shalt thou see\
3840 Lorenzo, who is thy new master's guest,\
3841 Give him this letter,---do it secretly,---\
3842 And so farewell: I would not have my father\
3843 See me \edtext{in}{
3844     \collation{Q; om. F.}}
3845         talk with thee.
3846 \pend
3847
3848 \speaker{Laun.}
3849 \edtext{}{\lemma{\textit{Laun.}}\collation{Q2; Clowne. Q, F.}}%
3850 \edtext{Adieu!}{
3851     \collation{\textnormal{Adiew}, Q, F.}}
3852 tears \edtext{exhibit}{
3853     \note{Eccles paraphrased "My tears serve to express what my
3854 tongue should, if sorrow would permit it," but probably it is
3855 Launce-\lot's blunder for prohibit (Halliwell) or inhibit
3856 (Clarendon).}}
3857 my tongue, most beautiful \edtext{pagan}{
3858     \note{This may have a scurrilous undertone: cf. \textit{2 H 4,}
3859 {\scf II.} \textrm{iii. 168.}}}%
3860 , most sweet \edtext{Jew!}{
3861     \collation{\textnormal{Iewe}, Q, F. \quad \textnormal{do}} Q, F;
3862         \textnormal{did} F2.}}%
3863 ---if a Christian \edtext{do}{
3864     \note{Malone upheld the reading of Qq and F by comparing {\scf II.}
3865 vi. 23: "When you shall please to play the thieves for
3866 wives"; Launcelot seems fond of hinting at what is going to
3867 happen (cf. {\scf II.} v. 22--3). If F2's "did" is accepted,
3868 \textit{get} is used for beget, as in {\scf III.} v. 9.}}
3869 not play the knave and get thee, I am much deceived; but \edtext{adieu!}{
3870     \collation{\textnormal{adiew}, Q, F.}}
3871 these \edtext{foolish drops do \edtext{something}{
3872     \collation{Q; \textnormal{somewhat} F.}}
3873 drown my manly spirit}{
3874     \lemma{foolish\textnormal{\dots}spirit}
3875     \note{"tears do not become a man" (\textit{AYL.}, {\scf III.}
3876 iv. 3); cf. also \textit{H 5,} {\scf IV.} vi. 28--32.}}%
3877 : \edtext{adieu!}{
3878     \collation{\textnormal{adiew}. Q, F. \quad \textnormal{S. D.}} Q2, F; om. Q;
3879     after l. 15 Capell.}}
3880 \exit{Exit.}
3881 \pend

```

```

3882
3883 \speaker{Jes.}
3884 Farewell good Launcelot.\
3885 Alack, what heinous sin is it in me\
3886 To be ashamed to be my father's \edtext{child!}{
3887 \collation{\textnormal{child}, Q, F; \textnormal{Child?} Rowe.}}
3888 \pend
3889 \endnumbering
3890
3891 \end{document}
3892
3893 \</arden>

```

A.5 Classical text edition

The next example, which was extracted from a longer file kindly supplied by Wayne Sullivan, University College, Dublin, Ireland, illustrates the use of `ledmac` to produce a Latin text edition, the *Periphyseon*, with Greek passages.³¹ The Greek font used is that prepared by Silvio Levy and described in *TUGboat*.³² The output of this file is shown in Figure 5. Note the use of two layers of footnotes to record testimonia and manuscript readings respectively.

I have converted the original EDMAC example file from TeX to something that looks more like LaTeX.

```

3894 \<periph>
3895 % ledmixed.tex
3896 \documentclass{article}
3897 \usepackage{ledmac}
3898
3899 \noendnotes
3900 %% \overfullrule0 pt
3901 \lefthyphenmin=3
3902

```

The LaTeX version uses the `lgreek` package to access Silvio Levy's greek font. The `delims` package option subverts³³ the normal meaning of `$` to switch in and out of math mode. We have to save the original meaning of `$` before calling the package. Later, we use `\Ma` and `\aM` for math mode switching.

```

3903 \let\Ma=$
3904 \let\aM=$
3905 \usepackage[delims]{lgreek}
3906
3907 % We need an addition to \no@expands since the \active $ in lgreek

```

³¹The bibliographic details of the forthcoming book are: Iohannis Scotti Erivgenae, *Periphyseon* (*De Divisione Naturae*) Liber Quartus [Scriptores Latini Hiberniae vol. xii], (Dublin: School of Celtic Studies, Dublin Institute for Advanced Studies, forthcoming 1992).

³²*TUGboat* 9 (1988), pp. 20–24.

³³It actually changes its category code.

```

3908 % causes problems:
3909 \newcommand{\morenoexpands}{\let$=0}
3910
3911 \makeatletter
3912
3913 \newbox\lp@rbox
3914
3915 \newcommand{\ffootnote}[1]{%
3916   \ifnumberedpar@
3917     \xright@appenditem{\noexpand\vffootnote{f}{\l@d@nums}{\@tag}{#1}}{%
3918                                           \to\inserts@list
3919     \global\advance\insert@count by 1
3920   % \else           %% may be used only in numbered text
3921   %   \vffootnote{f}{\{0/0/0/0/0/0/0\}{#1}}%
3922   \fi\ignorespaces}
3923
3924 \newcommand{\gfootnote}[1]{%
3925   \ifnumberedpar@
3926     \xright@appenditem{\noexpand\vgfootnote{g}{#1}}{%
3927                                           \to\inserts@list
3928     \global\advance\insert@count by 1
3929   % \else           %% may be used only in numbered text
3930   %   \vgfootnote{g}{#1}%
3931   \fi\ignorespaces}
3932
3933 \newcommand{\setlp@rbox}[3]{%
3934   {\parindent\z@\hspace=2.5cm\raggedleft\scriptsize
3935     \baselineskip 9pt%
3936     \global\setbox\lp@rbox=\vbox to\z@{\vss#3}}}
3937
3938 \newcommand{\vffootnote}[2]{\setlp@rbox#2}
3939
3940 \newcommand{\vgfootnote}[2]{\def\rd@ta{#2}}
3941
3942
3943
3944 \renewcommand{\affixline@num}{%
3945   \ifsublines@
3946     \@l@tempcntb=\subline@num
3947     \ifnum\subline@num>\c@firstsublinenum
3948       \@l@tempcnta=\subline@num
3949       \advance\@l@tempcnta by-\c@firstsublinenum
3950       \divide\@l@tempcnta by\c@sublinenumincrement
3951       \multiply\@l@tempcnta by\c@sublinenumincrement
3952       \advance\@l@tempcnta by\c@firstsublinenum
3953     \else
3954       \@l@tempcnta=\c@firstsublinenum
3955     \fi
3956     %
3957     \ifcase\sub@lock

```

```

3958     \or
3959         \ifnum\sublock@disp=1
3960             \@l@tempcntb=0 \@l@tempcnta=1
3961         \fi
3962     \or
3963         \ifnum\sublock@disp=2 \else
3964             \@l@tempcntb=0 \@l@tempcnta=1
3965         \fi
3966     \or
3967         \ifnum\sublock@disp=0
3968             \@l@tempcntb=0 \@l@tempcnta=1
3969         \fi
3970 \fi
3971 \else
3972     \@l@tempcntb=\line@num
3973     \ifnum\line@num>\c@firstlinenum
3974         \@l@tempcnta=\line@num
3975         \advance\@l@tempcnta by-\c@firstlinenum
3976         \divide\@l@tempcnta by\c@linenumincrement
3977         \multiply\@l@tempcnta by\c@linenumincrement
3978         \advance\@l@tempcnta by\c@firstlinenum
3979     \else
3980         \@l@tempcnta=\c@firstlinenum
3981     \fi
3982 \ifcase\@lock
3983     \or
3984         \ifnum\lock@disp=1
3985             \@l@tempcntb=0 \@l@tempcnta=1
3986         \fi
3987     \or
3988         \ifnum\lock@disp=2 \else
3989             \@l@tempcntb=0 \@l@tempcnta=1
3990         \fi
3991     \or
3992         \ifnum\lock@disp=0
3993             \@l@tempcntb=0 \@l@tempcnta=1
3994         \fi
3995 \fi
3996 \fi
3997 %
3998 \ifnum\@l@tempcnta=\@l@tempcntb
3999     \@l@tempcntb=\line@margin
4000     \ifnum\@l@tempcntb>1
4001         \advance\@l@tempcntb by\page@num
4002     \fi
4003     \ifodd\@l@tempcntb
4004 %         #1\rlap{{\rightlinenum}}}%
4005         \xdef\rd@ta{\the\line@num}%
4006     \else
4007         \llap{{\leftlinenum}}}%#1%

```



```

4008     \fi
4009     \else
4010         %#1%
4011     \fi
4012     \ifcase\@lock
4013     \or
4014         \global\@lock=2
4015     \or \or
4016         \global\@lock=0
4017     \fi
4018     \ifcase\sub@lock
4019     \or
4020         \global\sub@lock=2
4021     \or \or
4022         \global\sub@lock=0
4023     \fi}
4024
4025 \lineation{page}
4026 \linenummargin{right}
4027 \footparagraph{A}
4028 \footparagraph{B}
4029
4030 \renewcommand{\notenumfont}{\footnotesize}
4031 \newcommand{\notetextfont}{\footnotesize}
4032
4033 \let\Afootnoterule=\relax
4034 \count\Afootins=825
4035 \count\Bfootins=825
4036
4037 \newcommand{\Aparafootfmt}[3]{%
4038     \ledsetnormalparstuff
4039     \scriptsize
4040     \notenumfont\printlines#1\enspace
4041     %      \lemmafont#1/#2\enskip
4042     \notetextfont
4043     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
4044
4045 \newcommand{\Bparafootfmt}[3]{%
4046     \ledsetnormalparstuff
4047     \scriptsize
4048     \notenumfont\printlines#1\enspace
4049     \select@lemmafont#1/#2\rbracket\enskip
4050     \notetextfont
4051     #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
4052 \makeatother
4053
4054 \let\Afootfmt=\Aparafootfmt
4055 \let\Bfootfmt=\Bparafootfmt
4056 \def\lemmafont#1/#2/#3/#4/#5/#6/#7/{\scriptsize}
4057 \parindent=1em

```

```

4058
4059 \newcommand{\lmarpar}[1]{\edtext{}{\ffootnote{#1}}}
4060 \newcommand{\rmarpar}[1]{\edtext{}{\gfootnote{#1}}}
4061 \emergencystretch40pt
4062
4063 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4064
4065 \begin{document}
4066
4067 \beginnumbering
4068 \pstart
4069 \rmarpar{741C}
4070 \noindent \edtext{Incipit Quartus $PERIFUSEWN$}{%
4071 \lemma{incipit\ .~.~.\ $PERIFUSEWN$}\Bfootnote{\textit{om.\ R},
4072 incipit quartus \textit{M}}}
4073 \pend
4074 \medskip
4075
4076 \pstart
4077 \noindent \edtext{NVTRITOR}{\lemma{$ANAKEFALIOSIS$}\Bfootnote{\textit{
4078 FJP, lege} $<anakefala'iwsis$}}.\lmarpar{$ANAKEFALIOSIS$
4079 NATVRARVM} Prima nostrae
4080 \edtext{Physiologiae}{\lemma{physiologiae}\Bfootnote{phisiologiae
4081 \textit{P}, physeologiae \textit{R}}}
4082 intentio praecipuaque mat\ -e\ -ria erat
4083 \edtext{quod}{\Bfootnote{\textit{p}.\ natura \textit{transp.\ MR}}}
4084 \edtext{$UPEROUSIADES$}{\Bfootnote{\textit{codd.\ Vtrum}
4085 $<uperousi'wdhs$ (hoc est superessentialis) natura \textit{cum Gale
4086 (p.160) an} $<uperousi'oths$ (hoc est superessentialis natura)
4087 \textit{cum Floss (PL 122,741C) intelligendum sit, ambigitur}}}
4088 (hoc est superessentialis) natura sit causa creatrix existentium et
4089 non existentium omnium, a nullo creata, unum principium, una
4090 origo, unus et uniuersalis uniuersorum fons, a nullo manans, dum
4091 ab eo manant omnia, trinitas coessentialis in tribus substantiis,
4092 $ANARQOS$ (hoc est sine principio), principium et finis, una
4093 bonitas, deus unus,
4094 \edtext{$OMOUSIOS$}{\Bfootnote{\textit{codd., lege} $<omoo'usios$}}
4095 \edtext{et}{\lemma{\textbf{et}}\Bfootnote{\textit{
4096 R}\textsuperscript{1}, \textit{om.\ R}\textsuperscript{0}}}
4097 $UPEROUSIOS$ (id est coessentialis et superessentialis). Et, ut
4098 ait sanctus Epifanius, episcopus Constantiae Cypri, in
4099 \edtext{$AGKURATW$}{\Bfootnote{anchurato \textit{MR}}}
4100 sermone
4101 \edtext{de fide}{\Bfootnote{Glo\Ma\langle\am ssa\Ma\rangle\am: Ita
4102 enim uocatur sermo eius de fide $AGKURATOS$, id est procuratus
4103 \textit{mg.\ add.\ FJP}}}:
4104 \begin{itshape}Tria sancta, tria consancta, tria
4105 \edtext{agentia}{\Bfootnote{actiua \textit{MR}}},
4106 tria coagentia, tria
4107 \edtext{formantia}{\Bfootnote{formatiua \textit{MR}}},

```

```

4108 tria conformantia, tria
4109 \edtext{operantia}{\Bfootnote{operatiua \textit{MR}}},
4110 tria cooperantia, tria subsistentia, tria\rm arpar{742C}
4111 consubsistentia sibi inuicem coexistentia. Trinitas haec
4112 sancta uocatur: tria existentia, una consonantia, una deitas
4113 \edtext{eiusdem}{\Bfootnote{eiusdemque \textit{M}}}}
4114 essentiae,
4115 \edtext{eiusdem uirtutis, eiusdem
4116 \edtext{subsistentiae}{\Bfootnote{substantiae \textit{R}}}}{\%
4117 \Bfootnote{\textit{om.\ M}}},
4118 similia
4119 \edtext{similiter}{\Bfootnote{ex simili \textit{MR}}}}
4120 aequalitatem gratiae operantur patris et filii et sancti spiritus.
4121 Quomodo autem
4122 \edtext{sunt}{\Bfootnote{\textit{om.\ M}}},
4123 ipsis relinquitur docere:
4124 \edtext{'Nemo enim nouit patrem nisi filius, neque filium nisi pater,
4125 et cuicumque filius reuelauerit'}{\Afootnote{Matth.\ 11, 27}};
4126 reuelatur autem per spiritum sanctum. Non ergo haec tria existentia
4127 aut ex ipso aut per ipsum aut ad ipsum in unoquoque digne intelliguntur,
4128 \Ma\mid\! R, 264^{\rm r}\!\mid\!aM\ sicut ipsa reuelant:\end{itshape}
4129 $FWS, PUR, PNEUMA$
4130 \edtext{(hoc est lux, ignis, spiritus)}{\Afootnote{EPIPHANIVS,
4131 \textit{Ancoratus} 67; PG~43, 137C--140A; GCS 25, p.~82, 2--12}}.
4132 \pend
4133
4134 \pstart
4135 Haec, ut dixi, ab Epifanio tradita, ut quisquis interrogatus quae
4136 tria et quid unum in sancta trinitate debeat credere, sana fide
4137 \Ma\!\mid J, 1^{\rm v}\!\mid\!aM\ respondere ualeat, aut ad
4138 fidem accedens\rm arpar{743A} sic erudiatur. Et mihi uidetur
4139 spiritum pro calore posuisse, quasi dixisset in similitudine:
4140 lux, ignis, calor. Haec enim tria unius essentiae sunt. Sed cur
4141 lucem primo dixit, non est mirum. Nam et pater lux est et ignis
4142 et calor; et filius est lux, ignis, calor; et
4143 \edtext{spiritus sanctus}{\Bfootnote{sanctus spiritus \textit{R}}}}
4144 lux, ignis, calor. Illuminat enim pater, illuminat filius, illuminat
4145 spiritus sanctus: ex ipsis enim omnis scientia et sapientia donatur.
4146 \pend
4147 \endnumbering
4148
4149 \end{document}
4150
4151 </periph>

```

A.6 Nijmegen

This example, illustrated in Figure 6, was provided in 2004 by Dirk-Jan Dekker of the Department of Medieval History at the University of Nijmegen³⁴. Unlike earlier examples, this was coded for LaTeX and ledmac from the start. I have reformatted the example to help it fit this document; any errors are those that I have inadvertently introduced. Note that repeated line numbers are eliminated from the footnotes.

```

4152 (*dekker)
4153 %% This is ledekker.tex, a sample critical text edition
4154 %% written in LaTeX2e with the ledmac package.
4155 %% (c) 2003--2004 by Dr. Dirk-Jan Dekker,
4156 %% University of Nijmegen (The Netherlands)
4157 %% (PRW) Modified slightly by PRW to fit the ledmac manual
4158
4159 \documentclass[10pt, letterpaper, oneside]{article}
4160 \usepackage[latin]{babel}
4161 \usepackage{ledmac}
4162
4163 \lineation{section}
4164 \linenummargin{left}
4165 \sidenotemargin{outer}
4166
4167 \renewcommand{\notenumfont}{\footnotesize}
4168 \newcommand{\notetextfont}{\footnotesize}
4169
4170 %\let\Afootnoterule=\relax
4171 %\let\Bfootnoterule=\relax
4172 %\let\Cfootnoterule=\relax
4173
4174 \addtolength{\skip\Afootins}{1.5mm}
4175 %\addtolength{\skip\Bfootins}{1.5mm}
4176 %\addtolength{\skip\Cfootins}{1.5mm}
4177
4178 \makeatletter
4179
4180 \renewcommand*{\para@vfootnote}[2]{%
4181   \insert\cename #1footins\endcename
4182   \bgroup
4183     \notefontsetup
4184     \footsplitskips
4185     \l@dparsefootspec #2\ledplinenumtrue % new from here
4186     \ifnum\@nameuse{previous@#1@number}=\l@dparsestartline\relax
4187       \ledplinenumfalse
4188       \fi
4189     \ifnum\previous@page=\l@dparsestartpage\relax
4190       \else \ledplinenumtrue \fi
4191     \ifnum\l@dparsestartline=\l@dparseendline\relax

```

³⁴On 1st September 2004 the University changed its name to Radboud University.

```

4192 \else \ledplinenumtrue \fi
4193 \expandafter\edef\csname previous@#1@number\endcsname{\l@dparsedstartline}
4194 \xdef\previous@page{\l@dparsedstartpage} % to here
4195 \setbox0=\vbox{\hsize=\maxdimen
4196 \noindent\csname #1footfmt\endcsname#2}%
4197 \setbox0=\hbox{\unvvh0}%
4198 \dp0=0pt
4199 \ht0=\csname #1footfudgefactor\endcsname\wd0
4200 \box0
4201 \penalty0
4202 \egroup
4203 }
4204
4205 \newcommand*\previous@A@number{-1}
4206 \newcommand*\previous@B@number{-1}
4207 \newcommand*\previous@C@number{-1}
4208 \newcommand*\previous@page{-1}
4209
4210 \newcommand{\abb}[1]{#1%
4211 \let\rbracket\nobrak\relax}
4212 \newcommand{\nobrak}{\textnormal{}}
4213 \newcommand{\morenoexpands}{%
4214 \let\abb=0%
4215 }
4216
4217 \newcommand{\Aparafootfmt}[3]{%
4218 \ledsetnormalparstuff
4219 \scriptsize
4220 \notenumfont\printlines#1\enspace
4221 % \lemmafont#1/#2\enskip
4222 \notetextfont
4223 #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
4224
4225 \newcommand{\Bparafootfmt}[3]{%
4226 \ledsetnormalparstuff
4227 \scriptsize
4228 \notenumfont\printlines#1%
4229 \ifledplinenum
4230 \enspace
4231 \else
4232 {\hskip 0em plus 0em minus .3em}
4233 \fi
4234 \select@lemmafont#1/#2\rbracket\enskip
4235 \notetextfont
4236 #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
4237
4238 \newcommand{\Cparafootfmt}[3]{%
4239 \ledsetnormalparstuff
4240 \notenumfont\printlines#1\enspace
4241 % \lemmafont#1/#2\enskip

```

```

4242 \notetextfont
4243 #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
4244
4245 \makeatother
4246
4247 \footparagraph{A}
4248 \footparagraph{B}
4249 \footparagraph{C}
4250
4251 \let\Afootfmt=\Aparafootfmt
4252 \let\Bfootfmt=\Bparafootfmt
4253 \let\Cfootfmt=\Cparafootfmt
4254
4255 \emergencystretch40pt
4256
4257 \author{Guillelmus de Berchen}
4258 \title{Chronicle of Guelders}
4259 \date{}
4260 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
4261 \begin{document}
4262 \maketitle
4263 \thispagestyle{empty}
4264
4265 \section*{St.\ Stephen's Church in Nijmegen}
4266 \beginnumbering
4267 \autopar
4268
4269 \noindent
4270 Nobilis itaque comes Otto imperio et dominio Novimagensi sibi,
4271 ut praeferetur, impignoratis et commissis
4272 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
4273 \textsc{liiii}\ledsidenote{1254} superius descripto, mense
4274 Iu\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis
4275 ceterisque civibus civitatis Novimagensis, pro ipsius et
4276 inhabitantium in ea necessitate,\edtext{}{\Afootnote{p.\ 97~N}}
4277 commodo et utilitate, ut
4278 \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}}
4279 parochialis
4280 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
4281 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
4282 \edtext{transfer\edtext{}{\Afootnote{p.\ 129~D}}retur}%
4283 {\Bfootnote{transferreretur NH}}
4284 ac de novo construeretur, \edtext{a reverendo patre domino
4285 \edtext{Conrado de \edtext{Hofsteden}}%
4286 {\Bfootnote{Hoffstede D: Hoffsteden H}},
4287 archiepiscopo
4288 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}}%
4289 {\Cfootnote{Conrad of Hochstaden was archbishop of Cologne in
4290 1238--1261}}, licentiam){\Cfootnote{William is confusing two
4291 charters that are five years apart. Permission from St.\ Apostles'

```

4292 Church in Cologne had been obtained as early as 1249. Cf.\ Sloet,
 4293 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
 4294 ‘‘\ldots{}nos devotionis tue precibus annuentes, ut ipsam
 4295 ecclesiam faciens demoliri transferas in locum alium competentem,
 4296 tibi auctoritate presentium indulgemus\ldots{}’’}, et a
 4297 venerabilibus \edtext{dominis}{\Bfootnote{viris H}} decano et
 4298 capitulo sanctorum Apostolorum
 4299 \edtext{Colonien}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab
 4300 antiquo veris et pacificis patronis, consensum, citra tamen
 4301 praeiudicium, damnum aut gravamen
 4302 \edtext{iurium}{\Bfootnote{virium D}} et bonorum eorundem,
 4303 impetravit.
 4304
 4305 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}} locum
 4306 eiusdem civitatis \edtext{qui}{\Bfootnote{quae D}} dicitur
 4307 \edtext{Hundisbrug}{\Bfootnote{Hundisburch D: Hunsdisbrug R}},
 4308 de praelibati Wilhelmi Romanorum
 4309 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
 4310 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
 4311 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
 4312 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
 4313 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de
 4314 expresso eiusdem civitatis assensu libera contradiderunt voluntate,
 4315 obligantes se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
 4316 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et
 4317 capitulo, quod in recompensationem illius areae infra castrum et
 4318 portam, quae fuit dos ecclesiae, in qua plebanus habitare
 4319 solebat---quae \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum
 4320 civitatis est destructa---aliam aream competentem et ecclesiae
 4321 novae,
 4322 \edtext{ut praefertur, aedificandae}{\lemma{\abb{ut\ldots aedificandae}}}%
 4323 \Bfootnote{\textit{om.}\ H}} satis
 4324 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent
 4325 et assignarent. Et desuper
 4326 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
 4327 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
 4328 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
 4329 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
 4330 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
 4331 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.}%
 4332 {\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762 (June 1254)}}
 4333
 4334 % (PRW) the full document continues on after this point
 4335 %%%
 4336 \endnumbering
 4337 \end{document}
 4338 %%%
 4339
 4340 </dekker>

A.7 Irish verse

This example, illustrated in Figure 7, is a somewhat modified and shortened version of Wayne Sullivan's example demonstration for EDSTANZA.

The stanza lines are numbered according to the source verse lines, not according to the printed lines. For example, the sixth ('f') line in the first stanza is printed as two lines as the source line was too long to fit on one printed line. Note that if you process this yourself you will get error reports about counters the first time through; this is because alphabetic counters, like roman numerals, have no notion of zero.

As is fairly typical of critical edition typesetting, some of ledmac's internal macros had to be modified to get the desired effects.

```

4341 (*braonain)
4342 %% This is ledbraonain.tex, a sample critical verse edition.
4343 %% Originally written for TeX processing with edmac and edstanza
4344 %% by Wayne Sullivan.
4345 %% Extensively modified by Peter Wilson for LaTeX and the ledmac package.
4346
4347 \documentclass{article}
4348 \usepackage{ledmac}
4349
4350 \setlength{\textheight}{40pc}
4351 \setlength{\textwidth}{24pc}
4352 \bigskipamount=12pt plus 6pt minus 6pt
4353 \newcommand*{\notetextfont}{\footnotesize}
4354
4355 %% Just one footnote series
4356 \footparagraph{C}
4357 \count\Cfootins=800
4358 \makeatletter
4359 %% but using two different formats
4360 \def\xparafootfmt#1#2#3{%
4361   \ledsetnormalparstuff
4362   {\notenumfont\printlines#1}\enspace
4363   %% {\select@lemmafnt#1/#2}\rbracket\enskip
4364   \notetextfont #3\penalty-10 }
4365 \def\yparafootfmt#1#2#3{%
4366   \ledsetnormalparstuff
4367   %% {\notenumfont\printlines#1}\enspace
4368   %% {\select@lemmafnt#1/#2}\rbracket\enskip
4369   \notetextfont #3\penalty-10 }
4370
4371 \let\Cfootfmt=\xparafootfmt
4372 \skip\Cfootins=\bigskipamount
4373 \makeatother
4374
4375 %% This is the default, but just to demonstrate...
4376 \setlength{\stanzaindentbase}{20pt}
4377

```



```

4378 %%                                MUST SET THE INDENTS
4379 %% indent multiples; first=hangindent.
4380 %% Must all be non-negative whole numbers
4381 \setstanzaindents{4,1,2,1,2,3,3,1,2,1}
4382
4383 %%                                Set stanza line penalties
4384 %% Must be nonnegative whole numbers.
4385 %% An initial zero indicates no penalties.
4386 \setstanzapenalties{1,5000,10500,5000,10500,5000,5000,5000,0}
4387 %\setstanzapenalties{0}% the default
4388
4389 %%                                Put some space between stanzas
4390 \let\endstanzaextra=\bigbreak % ==> \bigskip \penalty -200
4391
4392 %% (almost) force line break in foot paragraph
4393 \mathchardef\IMM=9999
4394 \def\lbreak{\hfil\penalty-\IMM}
4395
4396 %%                                Number each stanza in bold
4397 \newcounter{stanzanum}
4398 \setcounter{stanzanum}{0}
4399 \newcommand*{\numberit}{%
4400   \flagstanza[0.5\stanzaindentbase]{\textbf{\thestanzanum}}}
4401 %% Use the hook to insert the number (and counteract a new line)
4402 %% and reset the line number to zero
4403 \newcommand*{\startstanzahook}{\refstepcounter{stanzanum}%
4404   \numberit\vskip-\baselineskip%
4405   \setlinenum{0}}
4406
4407 %% Want to label the footnotes with the stanza and line number
4408 %% We'll use \linenum to replace the sub-line number
4409 %% with the stanza number, redefining \edtext to do this
4410 %% automatically for us.
4411 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4412 \makeatletter
4413
4414 \renewcommand{\edtext}[2]{\leavevmode
4415   \begingroup
4416     \noexpands
4417     \xdef\@tag{#1}%
4418     \set@line
4419     \global\insert@count=0
4420     \ignorespaces \linenum{||\the\c@stanzanum}#2\relax
4421     \flag@start
4422   \endgroup
4423   #1%
4424   \ifx\end@lemmas\empty \else
4425     \glp\end@lemmas\to\x@lemma
4426     \x@lemma
4427     \global\let\x@lemma=\relax

```

```

4428 \fi
4429 \flag@end}
4430
4431 %% We need only a very simple macro for footnote numbers,
4432 %% to produce the stanza number (sub-line) then the line number.
4433 \def\printstanzalines#1/#2/#3/#4/#5/#6/#7/{\begingroup
4434   #3\fullstop \linenumrep{#2}
4435 \endgroup}
4436 \let\oldprintlines\printlines
4437
4438 \makeatother
4439 %%%%%%%%%%%%%%%
4440
4441 \pagestyle{empty}
4442
4443 \begin{document}
4444
4445 \beginnumbering
4446
4447 \pstart \centering \textbf{22} \pend
4448
4449 \bigskip
4450 %% do not print line number beside heading
4451 \setcounter{firstlinenum}{1000}
4452 %% and heading footnotes use a different format
4453 \let\Cfootfmt=\yparafootfmt
4454
4455 \pstart
4456 \centerline{[Se\'an \'O Braon\'ain cct] chuim Tom\'ais U\'{\i}
4457 \edtext{Dh\'unlaing}{\Cfootnote{\textbf{22} \textit{Teideal}: Dhuinnluinnng T,
4458 Se\'aghan Mac Domhnaill cct B\lbreak}}}
4459 \pend
4460
4461 \pstart
4462 \centerline{[Fonn: M\'airse\'ail U\'{\i} Sh\'uilleabh\'ain (P\'ainseach
4463           na nUbh]}
4464 \pend
4465
4466 \bigskip
4467
4468 %%           revert to the regular footnote format
4469 \let\Cfootfmt=\xparafootfmt
4470 %%           but use our special number printing routine
4471 \let\printlines\printstanzalines
4472 %%           Use letters for line numbering
4473 \linenumberstyle{alph}
4474 %%           number lines from the second onwards
4475 \setcounter{firstlinenum}{2}
4476 \setcounter{linenumincrement}{1}
4477

```

```

4478 %% Each verse starts with \stanza.
4479 %% Lines end with &; the last line with \&.
4480
4481 \stanza
4482 A \edtext{dhuine}{\Cfootnote{dhuinne T}} gan ch\’eill do
4483 \edtext{mhaisligh}{\Cfootnote{mhaslaidh T, mhaslaig B}} an chl\’eir&
4484 is tharcaisnigh naomhscruipt na bhf\’aige,&
4485 na haitheanta \edtext{r\’eab}{\Cfootnote{raob T}} ’s an
4486 t-aifreann thr\’eig&
4487 \edtext{re}{\Cfootnote{le B}} taithneamh do chlaonchreideamh
4488 Mh\’artain,&
4489 c\’a rachair \edtext{od}{\Cfootnote{dod B}} dh\’\{i\}on ar
4490 \’Iosa Nasardha&
4491 nuair \edtext{chaithimid}{\Cfootnote{chaithfam\’\{i\}d T}} cruinn
4492 bheith ar \edtext{mhaoileann}{\Cfootnote{maoilinn B}} Josepha?&
4493 N\’\{i\} caraid Mac Crae chuim t’anama ’
4494 \edtext{phl\’e}{\Cfootnote{phleideh T}}&
4495 n\’a Calvin \edtext{bhiais}{\Cfootnote{bh\’\{i\}os B}} taobh
4496 \edtext{ris}{\Cfootnote{leis B}} an l\’a sin.\&
4497
4498 \stanza
4499 N\’ach damanta an sc\’eal don chreachaire
4500 \edtext{chlaon}{\Cfootnote{claon B}}&
4501 ghlac baiste na cl\’eire ’na ph\’aiste&
4502 ’s do \edtext{glanadh}{\Cfootnote{glannuig T}} mar ghr\’ein
4503 \’on bpeaca r\’o-dhaor&
4504 tr\’\{i\} \edtext{ainibhfios}{\Cfootnote{ainnibhfios T, ainnbhfios B}}
4505 \edtext{\’Eva}{\Cfootnote{\’Eabha B}} rinn \’Adam,&
4506 tuitim ar\’\{i\}s f\’e chuing na haicme sin&
4507 tug atharrach br\’\{i\} don scr\’\{i\}bhinn bheannaithe,&
4508 d’aistrigh b\’easa \edtext{agus}{\Cfootnote{is B}} reachta na cl\’eire&
4509 ’s n\’ach \edtext{tugann}{\Cfootnote{tuigionn T}} aon
4510 gh\’eilleadh don Ph\’apa?\&
4511
4512 \stanza
4513 Gach \edtext{scolaire}{\Cfootnote{sgollaire T}} baoth, n\’\{i\}
4514 \edtext{mholaim}{\Cfootnote{mholluim T}} a cheird&
4515 \edtext{t\’a ag obair}{\Cfootnote{t\’ag ccobar T}}
4516 \edtext{le}{\Cfootnote{re B}} g\’eilleadh d\’a th\’aille&
4517 don \edtext{doirbhchoin chlaon}{\Cfootnote{dorbhchon daor B}}
4518 d\’a ngorthar Mac Crae,&
4519 deisceabal \edtext{straeigh}{\Cfootnote{straodhaig T}} as an
4520 gcoll\’aiste.&
4521 T\’a \edtext{\edtext{adaithe}}{\Cfootnote{fadaighthe B}}
4522 th\’\{i\}os{\Cfootnote{fhadoghthe ts\’\{i\}os T}} in
4523 \’\{i\}ochtar ifrinn,&
4524 gan \edtext{solas}{\Cfootnote{sollus T}} gan soilse i
4525 dt\’\{i\}orthaibh dorchas,&
4526 tuigsint an l\’einn, gach
4527 \edtext{cuirpeacht}{\Cfootnote{cuirripeacht T}} d\’ein&

```

```
4528 is \edtext{Lucifer}{\Cfootnote{Luicifer T, L\'ucifer B}} aosta
4529 'na \edtext{mh\'aistir}{\Cfootnote{mhaighistir T}}.\&
4530
4531 \endnumbering
4532
4533 \end{document}
4534
4535 </braonain>
```

References

- [Bre96] Herbert Breger. **TABMAC**. October 1996. (Available from CTAN in `macros/plain/contrib/tabmac`)
- [Bur01] John Burt. ‘Typesetting critical editions of poetry’. *TUGboat*, **22**, 4, pp 353–361, December 2001. (Code available from CTAN in `macros/latex/contrib/poemscol`)
- [Eck03] Matthias Eckermann. *The Parallel-Package*. April 2003. (Available from CTAN in `macros/latex/contrib/parallel`)
- [Fai03] Robin Fairbairns. *footmisc — a portmanteau package for customising footnotes in LaTeX*. February 2003. (Available from CTAN in `macros/latex/contrib/footmisc`)
- [LW90] John Lavagnino and Dominik Wujastyk. ‘An overview of EDMAC: a PLAIN TeX format for critical editions’. *TUGboat*, **11**, 4, pp. 623–643, November 1990. (Code available from CTAN in `macros/plain/contrib/edmac`)
- [Lüc03] Uwe Lück. ‘ednotes — critical edition typesetting with LaTeX’. *TUGboat*, **24**, 2, pp. 224–236, 2003. (Code available from CTAN in `macros/latex/contrib/ednotes`)
- [Sul92] Wayne G. Sullivan. *The file edstanza.doc*. June 1992. (Available from CTAN in `macros/plain/contrib/edmac`)
- [Wil02] Peter Wilson. *The memoir class for configurable typesetting*. November 2002. (Available from CTAN in `macros/latex/contrib/memoir`)
- [Wil04] Peter Wilson. *Parallel typesetting for critical editions: the ledpar package*. December 2004. (Available from CTAN in `macros/latex/contrib/ledmmac`)
- [Wil05] Peter Wilson. *Critical editions and arabic typesetting: the ledarab and afoot packages*. February 2005. (Available from CTAN in `macros/latex/contrib/ledmmac`)

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	
<code>\&</code>	20, 2703, 2707, 2708, 2757, 2766, 2772, 2774, 4479, 4496, 4510, 4529
<code>\-</code>	3855, 4082
<code>\@@line</code>	1523
<code>\@wrindexm@m</code>	2630, 2635, 2638, 2662
<code>\@EDROWFILL@</code>	2995, <u>3202</u>
<code>\@M</code>	1523, 2751, 2761
<code>\@MM</code>	1268
<code>\@adv</code>	<u>458</u> , 619
<code>\@arabic</code>	749
<code>\@aux</code>	1803
<code>\@auxout</code>	1805, 2629, 2634, 2637, 2661
<code>\@botlist</code>	1756, 1758
<code>\@ccclv</code>	1651, 1655, 1656, 1754, 1755, 1783
<code>\@checkend</code>	2690
<code>\@colht</code>	1633, 1759, 1771
<code>\@colroom</code>	1759
<code>\@combinefloats</code>	1628
<code>\@currenvir</code>	2675, 2678, 2679
<code>\@currlist</code>	1760, 1763
<code>\@dbldeferlist</code>	1769, 1774, 1776
<code>\@dblfloatplacement</code>	1773
<code>\@dbltoplist</code>	1769, 1770
<code>\@deferlist</code>	1756, 1765, 1766
<code>\@docclearpage</code>	<u>1738</u>
<code>\@edrowfill@</code>	<u>3202</u>
<code>\@ehb</code>	1762
<code>\@emptytoks</code>	<u>2666</u> , 2676
<code>\@footnotemark</code>	<u>2114</u>
<code>\@footnotemarkA</code>	2147, 2159
<code>\@footnotemarkB</code>	2431, 2442
<code>\@footnotemarkC</code>	2449, 2458
<code>\@footnotetext</code> <u>2110</u> , <u>2127</u> , 2532, 2558, 2585
<code>\@freelist</code>	1626
<code>\@gobble</code>	653–657, 1925, 2687, 2964, 2981
<code>\@gobblethree</code>	<u>1917</u>
<code>\@h</code>	<u>1521</u>
<code>\@hilfs@count</code>	<u>2897</u>
<code>\@idxfile</code>	2620, 2630, 2635, 2638, 2662
<code>\@ifclassloaded</code> 29, 1701, 1726, 2109, 2607
<code>\@ifnextchar</code>	2610, 2877
<code>\@ifpackageloaded</code>	2658
<code>\@iiiminipage</code>	<u>2521</u>
<code>\@iiiparbox</code>	2548
<code>\@indexfile</code>	2652
<code>\@inputcheck</code>	345
<code>\@insert</code>	1121–1123, 1157–1159
<code>\@k</code>	<u>1521</u>
<code>\@kludgeins</code>	1630, 1698
<code>\@l</code>	<u>375</u> , 602
<code>\@l@dttempcnta</code>	... <u>27</u> , 492, 494, 496, 497, 905, 906, 908, 910, 913, 914, 929, 965–969, 971, 978– 982, 984, 987, 990, 992, 996, 1026, 1030, 1034, 1041, 1045, 1049, 1130, 1134, 1138, 1141, 1144, 1147, 1148, 3948–3952, 3954, 3960, 3964, 3968, 3974– 3978, 3980, 3985, 3989, 3993, 3998
<code>\@l@dttempcntb</code> <u>27</u> , 212, 213, 218, 222, 226, 230, 233, 256, 257, 264, 268, 272, 274, 282, 283, 963, 975, 996, 1004–1006, 1008, 1026, 1030, 1034, 1041, 1045, 1049, 1079–1081, 1083, 1136, 1137, 1983, 1984, 1989, 1993, 1997, 2001, 2004, 2083–2085, 2087, 3946, 3960, 3964, 3968, 3972, 3985, 3989, 3993, 3998–4001, 4003
<code>\@l@reg</code>	<u>375</u>
<code>\@lab</code>	575, 1794, <u>1840</u>
<code>\@latexerr</code>	1762
<code>\@led@extranofeet</code>	... <u>1723</u> , 1736, 1749
<code>\@led@nofootfalse</code>	1740–1748
<code>\@led@nofoottrue</code>	1739
<code>\@led@testifnofoot</code>	<u>1738</u>
<code>\@line@num</code>	<u>2893</u> , 3007
<code>\@listdepth</code>	2534
<code>\@lock</code>	141, <u>333</u> , 400, 402, 404, 417, 525, 526, 528, 529, 545, 546, 548, 876,

- 935, 937, 938, 940, 1038, 1053,
1055, 1057, 3982, 4012, 4014, 4016
\@lopL 442
\@lopR 442
\@makecol 1705
\@makefcolumn .. 1765, 1766, 1774, 1776
\@makespecialcolbox 1631
\@maxdepth 1646, 1654
\@mem@extranofeet 1727
\@mem@nofootfalse 1728–1735
\@midlist 1626, 1627
\@minipagefalse 2545
\@minipagerestore 2535
\@minus 1398, 1403, 2252, 2258
\@mpargs 2525, 2548
\@mpfn 2531, 2557, 2584
\@mpfootins 2541, 2551, 2554, 2563, 2590
\@mpfootnotetext ... 2532, 2558, 2585
\@mplistdepth 2534
\@nameuse 289,
291, 1273, 1274, 1379, 1381,
1452, 1453, 1493, 1495, 1571,
1573, 1614, 1616, 1712, 1713,
1715, 1716, 1718, 1720, 2176,
2180, 2182, 2186, 2189, 2190,
2195, 2199, 2203, 2206, 2209,
2214, 2217, 2219, 2220, 2224,
2231, 2237, 2282, 2292, 2300,
2302, 2327, 2337, 2345, 2347,
2373, 2377, 2378, 2386, 2394,
2395, 2399, 2409, 2419, 2421,
2485, 2486, 2488, 2489, 2494, 4186
\@nobreakfalse 764
\@nobreaktrue 762, 766
\@nowrindex 2619
\@oldnobreak 762, 764, 799
\@opcol 1766, 1784
\@opxtrafeetii 1675, 1676, 1711
\@outputbox
. 1633, 1635, 1636, 1651, 1653,
1673, 1674, 2467, 2468, 2483, 2484
\@outputpage 1775
\@page 422
\@parboxrestore 1277, 2193, 2530
\@pboxswfalse 2523
\@pend 442
\@pendR 442
\@plus 1283,
1398, 1403, 2252, 2258, 2288, 2333
\@ref 560, 605
\@ref@reg 562
\@reinserts 1706
\@set 473, 624
\@setminipage 2536
\@showidx 2627, 2660
\@tag 665, 683, 724, 1176,
1184, 1192, 1200, 1208, 1223,
1231, 1239, 1247, 1255, 1899,
1903, 1907, 1911, 1915, 3917, 4417
\@tempboxa 1754, 1755, 2526, 2548
\@tempdima 1655, 2524, 2528
\@templ@d 2074, 2075
\@textbottom 1638
\@texttop 1634
\@thefnmarkA 2146, 2158
\@thefnmarkB 2430, 2441
\@thefnmarkC 2448, 2457
\@toksa 316, 324
\@toksb 316, 323–325
\@toplist 1756, 1757
\@whilenum 3176
\@whilesw 1766, 1775
\@wredindex 2649, 2651, 2884
\@x@s f 2103, 2106, 2117, 2123, 2166, 2172
\@xloop 1155, 1162
\@xympar 1971
\^ 369, 2782

\l 3210, 3215, 3223, 3556,
3591, 3714, 3717, 4071, 4083,
4084, 4096, 4103, 4117, 4122,
4125, 4128, 4137, 4265, 4274,
4276, 4282, 4291–4293, 4310–
4312, 4316, 4323, 4326, 4329, 4332

A
\A@@footnote ... 2888, 2966, 2983, 3002
\abb 4210,
4214, 4280, 4311, 4316, 4322, 4326
\absline@num ... 138, 332, 380, 383,
386, 487, 490, 499, 513, 535,
557, 567, 866, 888, 889, 897, 1120
Abu Kamil Shuja' b. Aslam 7
\actionlines@list
..... 335, 352, 355, 362, 487,
490, 499, 513, 535, 557, 918, 921
\actions@list
. 335, 356, 363, 488, 497, 501,
503, 515, 524, 537, 544, 558, 922

- `\add@inserts` 839, 1109
`\add@inserts@next` 1109
`\add@penalties` 848, 1130
`\addfootins` 1709
`\addfootinsX` 2480
`\addtocounter` 796
`\addtol@denvbody` ... 2670, 2691, 2693
`\addtolength` 4174–4176
Adelard II 7
`\advancelabel@refs` 1800, 1809
`\advanceline` . 13, 63, 66, 619, 642, 657
`\advancepageno` 1621
`\Aend` 1897, 1917
`\Aendnote` 14, 1896
`\affixline@num` 837, 957, 3944
`\affixpstart@num` 838, 1068
`\affixside@note` 839, 2073
`\Afootfmt`
3657, 3794, 3808, 3827, 4054, 4251
`\Afootgroup` 1677
`\Afootins` 1386, 1677, 1688,
1728, 1741, 3658–3660, 4034, 4174
`\Afootnote` 14,
1173, 2500, 2888, 2958, 2966,
2975, 2983, 3002, 3451, 3458,
3544, 3547, 3549, 3580, 3582,
3655, 3684, 3710, 3712–3714,
3717, 3728, 3796, 4125, 4130,
4274, 4276, 4282, 4310, 4312, 4329
`\Afootnoterule` 3763, 4033, 4170
`\Afootstart` 1677, 3661
`\allowbreak`
1565, 1604, 2293, 2338, 3502, 3515
`\alph` 3428
`\alpha` 3549, 3584
`\aM` 3904, 4101, 4128, 4137
`\ampersand` 21, 2703, 2774
`\Aparafootfmt` . 4037, 4054, 4217, 4251
`\AtBeginDocument` 1836, 2658
`\author` 3435, 4257
`\autopar` 10, 84, 806, 4267
`\autoparfalse` 167, 807
`\autopartrue` 820

B

`\B@@footnote` ... 2888, 2967, 2984, 3003
`\ballast` 36
`\ballast@count` ... 883, 886, 891, 1130
Beeton, Barbara Ann Neuhaus Friend 10

`\beginnumbering`
... 8, 121, 180, 769, 817, 3446,
3540, 3682, 3810, 4067, 4266, 4445
`\beginnumberingR` 812
`\Bend` 1901, 1917
`\Bendnote` 14, 1896
`\beta` 3549, 3584
`\Bfootfmt` 3795, 4055, 4252
`\Bfootgroup` 1678
`\Bfootins` 1386,
1678, 1689, 1729, 1742, 4035, 4175
`\Bfootnote` ... 14, 1181, 2501, 2889,
2959, 2967, 2976, 2984, 3003,
3452, 3551, 3584, 3586, 3594,
3596, 3598, 3600, 3602, 3604,
3606, 3608, 3610, 3797, 4071,
4077, 4080, 4083, 4084, 4094,
4095, 4099, 4101, 4105, 4107,
4109, 4113, 4116, 4117, 4119,
4122, 4143, 4272, 4278, 4280,
4281, 4283, 4286, 4288, 4297,
4299, 4302, 4305–4307, 4309,
4311, 4313, 4315, 4316, 4319,
4323, 4324, 4326–4328, 4330, 4331
`\Bfootnoterule` 3763, 4171
`\Bfootstart` 1678
`\bfseries` 749, 3526
`\bigbreak` 4390
`\bigskip` . 3819, 3825, 4390, 4449, 4466
`\bigskipamount` 4352, 4372
`\body` 1163, 1164, 2705, 2773
`\bodyfootmarkA` 26
`\box` 857, 859, 1447, 1461, 1507, 1522,
1651, 1755, 1783, 2390, 2403, 4200
`\boxmaxdepth` 1654
`\Bparafootfmt` . 4045, 4055, 4225, 4252
Bredon, Simon 7
Breger, Herbert 5, 8, 152
Brey, Gerhard 7
`\brokenpenalty` 802
Burt, John 6
Busard, Hubert L. L. 7
`\bypage@false` 183, 196, 201
`\bypage@true` 183, 191
`\bypstart@false` 183, 192, 202
`\bypstart@true` 183, 197

C

`\C@@footnote` ... 2888, 2968, 2985, 3004
`\c@addcolcount` 3170

- \c@ballast 883, 891
 \c@firstlinenum ... 239, 977, 979,
 982, 984, 3973, 3975, 3978, 3980
 \c@firstsublinenum . 243, 964, 966,
 969, 971, 3947, 3949, 3952, 3954
 \c@footnoteA 2151
 \c@footnoteB 2434
 \c@footnoteC 2451
 \c@labidx 2596
 \c@linenumincrement
 239, 980, 981, 3976, 3977
 \c@mpfootnote 2531, 2557, 2584
 \c@page 602
 \c@pstart 749
 \c@stanzaanum 4420
 \c@sublinenumincrement
 243, 967, 968, 3950, 3951
 \Cend 1905, 1917
 \Cendnote 14, 1896
 \centering 4447
 \centerline 3106,
 3111, 3117, 3122, 3128, 3133,
 3354, 3356, 3812, 3822, 4456, 4462
 \Cfootfmt 4253, 4371, 4453, 4469
 \Cfootgroup 1679
 \Cfootins 1387, 1679,
 1690, 1730, 1743, 4176, 4357, 4372
 \Cfootnote 14, 1181,
 2502, 2890, 2960, 2968, 2977,
 2985, 3004, 3557, 3592, 4289,
 4290, 4332, 4457, 4482, 4483,
 4485, 4487, 4489, 4491, 4492,
 4494–4496, 4500, 4502, 4504,
 4505, 4508, 4509, 4513–4517,
 4519, 4521, 4522, 4524, 4527–4529
 \Cfootnoterule 4172
 \Cfootstart 1679
 \ch@ck@l@ck 994, 1022
 \ch@cksub@l@ck 973, 1022
 \changes 17
 \char 2703
 \chardef 1970, 2705, 2707
 Chester, Robert of 7
 Claassens, Geert H. M. 7
 class 1 feet 114, 139, 140
 class 2 feet 114, 115
 \cleaders 3165
 \closeout 594, 598, 1888
 \clubpenalty 802, 1134
 \collation 3796, 3814,
 3816, 3825, 3832, 3844, 3849,
 3851, 3861, 3870, 3872, 3878, 3887
 \color@begingroup
 1278, 1457, 1658, 2194, 2399, 2527
 \color@endgroup
 1279, 1457, 1662, 2195, 2399, 2546
 \columnwidth
 1276, 1428, 2192, 2529, 2572
 Copernicus, Nicolaus 7
 \copy 3670
 \count 1396, 1401, 1417,
 1421, 1545, 1548, 1589, 1611,
 2250, 2255, 2271, 2274, 2317,
 2320, 2358, 2361, 4034, 4035, 4357
 \countdef 1621
 \Cparafootfmt 4238, 4253
 \cr 1524, 1527
 \CRITEXT 2868
 \critext 39, 658, 664, 2870, 2957, 2999
 \cs 17, 702–704, 706–710, 845, 1809–1811
 \ctab 2788, 3313, 3404
 \ctabtext 2792, 3323, 3408
- D**
- \D@@footnote ... 2888, 2969, 2986, 3005
 \date 3436, 4259
 \dcolerr 2844, 2856
 \dcoli ... 2814, 2846, 2847, 3214, 3219
 \dcolii 2815, 2847
 \dcoliii 2816, 2847
 \dcoliv 2817, 2848
 \dcolix 2822, 2849
 \dcolv 2818, 2848
 \dcolvi 2819, 2848
 \dcolvii 2820, 2849
 \dcolviii 2821, 2849
 \dcolx 2823, 2849
 \dcolxi 2824, 2850
 \dcolxii 2825, 2850
 \dcolxiii 2826, 2850
 \dcolxiv 2827, 2851
 \dcolxix 2832, 2852
 \dcolxv 2828, 2851
 \dcolxvi 2829, 2851
 \dcolxvii 2830, 2852
 \dcolxviii 2831, 2852
 \dcolxx 2833, 2852
 \dcolxxi 2834, 2853
 \dcolxxii 2835, 2853

- \dcolxxiii 2836, 2853
 - \dcolxxiv 2837, 2854
 - \dcolxxix 2842, 2855
 - \dcolxxv 2838, 2854
 - \dcolxxvi 2839, 2854
 - \dcolxxvii 2840, 2855
 - \dcolxxviii 2841, 2855
 - \dcolxxx 2843, 2855
 - \DeclareOption 6, 7
 - \def@tempb 199
 - Dekker, Dirk-Jan 6, 31–33, 37, 196
 - \Dend 1909, 1917
 - \Dendnote 14, 1896
 - \Dfootgroup 1680
 - \Dfootins 1387, 1680, 1691, 1731, 1744
 - \Dfootnote 14, 1181, 2503, 2891, 2961,
2969, 2978, 2986, 3005, 3554, 3589
 - \Dfootstart 1680
 - \dimen 610, 611, 613–615, 617, 1397,
1402, 1426–1428, 1431, 1529–
1531, 1546, 1549, 1590, 1612,
2251, 2256, 2257, 2272, 2275,
2318, 2321, 2365–2367, 2370, 3658
 - \dimen@ 1635, 1637
 - \disablel@dtabfeet
..... 3020, 3037, 3051, 3065,
3078, 3093, 3221, 3228, 3233,
3241, 3246, 3254, 3272, 3288, 3411
 - \displaystyle 2907, 3022,
3024, 3053, 3055, 3080, 3082,
3221, 3233, 3246, 3334, 3386, 3387
 - \displaywidowpenalty 803
 - \divide 967,
980, 1428, 1530, 2367, 3950, 3976
 - \do@actions 868, 895
 - \do@actions@fixedcode 915, 928
 - \do@actions@next 895
 - \do@ballast 869, 883
 - \do@line 790, 828
 - \do@linehook 832, 850
 - \do@lockoff 534
 - \do@lockoffL 534
 - \do@lockon 505
 - \do@lockonL 505
 - \documentclass 3422,
3478, 3618, 3742, 3896, 4159, 4347
 - \doedindexlabel 2601, 2621, 2646, 2881
 - \doendnotes 22, 1963
 - \doreinxtrafeeti ... 1684, 2466, 2487
 - \doreinxtrafeetii .. 1685, 1687, 1714
 - \dosplits 1521
 - \dots 2786, 3556, 3591, 3714, 3717, 3874
 - \doublecolon 3505, 3514
 - Downes, Michael 36, 102, 104
 - \doxtrafeet 1668
 - \doxtrafeeti 1669, 2466, 2482
 - \doxtrafeetii 1670, 1672
 - \dp 1269, 1445,
1459, 1635, 1655, 2388, 2401, 4198
 - \dummy@edtext 646, 659
 - \dummy@ref 561, 571
 - \dummy@text 645, 658
- E**
- \E@footnote ... 2888, 2970, 2987, 3006
 - \edaftertab
.. 30, 165, 2794, 3261, 3295, 3313
 - edarrayc (environment) 27, 3403
 - edarrayl (environment) 27, 3403
 - edarrayr (environment) 27, 3403
 - \EDATAB 3353, 3361
 - \edatab 2795, 3359
 - \edatabell 2796, 3343
 - \edatleft 29, 2797, 3145
 - \edatright 29, 2798, 3153
 - \edbeforetab
.. 30, 165, 2793, 3261, 3295, 3313
 - \edfilldimen
..... 3167, 3177, 3178, 3180, 3204
 - \edfont@info 715, 718, 722
 - \EDINDEX 2874
 - \edindex 27, 2608, 2644, 2874,
2963, 2971, 2980, 2988, 3001,
3013, 3030, 3046, 3060, 3073, 3088
 - \edindexlab 27, 2596, 2602, 2605
 - \EDLABEL 2872
 - \edlabel 23, 653, 1793, 2602,
2872, 2991, 3010, 3012, 3029,
3045, 3059, 3072, 3087, 3220,
3227, 3232, 3240, 3245, 3253, 3459
 - \edmakelabel 24, 1883
 - \edpageref 23, 654, 1844
 - \edrowfill . 28, 2801, 2992, 2995, 3202
 - \EDTAB 3349, 3385
 - \edtabcolsep 28, 2936,
3025, 3042, 3055, 3069, 3083,
3098, 3178, 3235, 3248, 3257, 3371
 - \EDTABINDENT 3366, 3379
 - \edtabindent 3209,
3213, 3218, 3229, 3242, 3255, 3375

- `\EDTABtext` 3393
`edtabularc` (environment) 27, 3407
`edtabularl` (environment) 27, 3407
`edtabularr` (environment) 27, 3407
`\EDTEXT` 2868
`\edtext` 13, 659, 680,
2021–2023, 2130, 2235, 2868,
2974, 3000, 3451, 3452, 3458,
3543, 3546, 3548, 3550, 3552,
3553, 3558, 3560, 3562, 3564,
3566, 3568, 3570, 3572, 3574,
3579, 3581, 3583, 3585, 3587,
3588, 3593, 3595, 3597, 3599,
3601, 3603, 3605, 3607, 3609,
3684, 3710, 3712–3714, 3727,
3812, 3815, 3823, 3831, 3837,
3843, 3849, 3850, 3852, 3857,
3860, 3863, 3869, 3871, 3877,
3886, 4059, 4060, 4070, 4077,
4080, 4083, 4084, 4094, 4095,
4099, 4101, 4105, 4107, 4109,
4113, 4115, 4116, 4119, 4122,
4124, 4130, 4143, 4272, 4274,
4276, 4278, 4280–4282, 4284,
4285, 4288, 4297, 4299, 4302,
4305–4307, 4309–4313, 4315,
4316, 4319, 4322, 4324, 4326–
4331, 4409, 4414, 4457, 4482,
4483, 4485, 4487, 4489, 4491,
4492, 4494–4496, 4500, 4502,
4504, 4505, 4508, 4509, 4513–
4517, 4519, 4521, 4524, 4527–4529
`\edvertdots` 30, 2800, 3164
`\edvertline` 30, 2799, 3162
`\Eend` 1913, 1917
`\Eendnote` 14, 1896
`\Efootgroup` 1681
`\Efootins` 1388, 1681, 1692, 1732, 1745
`\Efootnote` 14, 1181,
2504, 2892, 2962, 2970, 2979,
2987, 3006, 3559, 3561, 3563,
3565, 3567, 3569, 3571, 3573, 3575
`\Efootstart` 1681
`\emergencystretch` 4061, 4255
`\empty` 25, 112, 154, 157, 314, 315, 352,
674, 690, 713, 727, 731, 737,
776, 918, 976, 992, 1111–1113,
1124, 1156, 1795, 2718, 3652, 4424
`\enablel@dtabfeet` 3269,
3285, 3303, 3311, 3321, 3329, 3411
`\end@lemmas`
644, 674, 675, 690, 691, 4424, 4425
`\endashchar` 18, 1289, 1365, 1955
`\endgraf` 787, 822, 826, 3645
`\endline@num` 340, 578, 584
`\endlock` 12, 634, 652, 2760, 2764, 2766
`\endminipage` 2538
`\endnumbering`
.. 8, 124, 147, 169, 179, 3461,
3613, 3735, 3889, 4147, 4336, 4531
`\endpage@num` 340, 577, 584
`\endprint` 22, 1917, 1966
`\endstanzaextra` 21, 2753, 4390
`\endsub` 12, 610, 651, 3666
`\endsubline@num` 340, 579, 585
`\enskip` ... 1287, 1483, 1564, 1603,
1918, 3489, 3501, 3514, 3521,
3650, 3653, 3773, 3779, 4041,
4049, 4221, 4234, 4241, 4363, 4368
`\enspace` 1286,
1482, 1563, 1602, 1918, 2199,
2292, 2337, 2409, 3488, 3513,
3520, 3750, 3778, 3787, 4040,
4048, 4220, 4230, 4240, 4362, 4367
environments:
`edarrayc` 27, 3403
`edarrayl` 27, 3403
`edarrayr` 27, 3403
`edtabularc` 27, 3407
`edtabularl` 27, 3407
`edtabularr` 27, 3407
`ledgroup` 22, 2556
`ledgroupsize` 22, 2569
`minipage` 22
Euclid 7
`\ExecuteOptions` 8
`\exit` 3752, 3880
`\extensionchars` 30, 110, 130, 175
- ## F
- `\f@encoding` 722
`\f@family` 722
`\f@series` 722
`\f@shape` 722
`\f@x@l@cks` 1017, 1022
Fairbairns, Robin 25
`\ffootnote` 3915, 4059
`\finishstage` 3665, 3666
`\first@linenum@out@false` ... 589, 595
`\first@linenum@out@true` 589

- `\firstlinenum` 10, 12, 248
 - `\firstpstart` 747
 - `\firstpstartfalse` 756
 - `\firstpstarttrue` 133, 752
 - `\firstsublinenum` 10, 12, 248
 - `\fix@page` 376, 429
 - `\flag@end` 603, 679, 695, 4429
 - `\flag@start` 603, 671, 687, 4421
 - `\flagstanza` 21, 2769, 4400
 - `\floatingpenalty` 1268
 - `\flush@notes` 792, 1154
 - Folkerts, Menso 7
 - `\fontencoding` 1170
 - `\fontfamily` 1170
 - `\fontseries` 1170
 - `\fontshape` 1170
 - `\footfootmarkA` 26, 2154
 - `\footfudgefiddle` 37, 1424, 1428, 2367
 - `\footgroupA` 2469
 - `\footgroupB` 2470
 - `\footgroupC` 2471
 - `\footins` . 1650, 1657, 1661, 1696, 1740
 - `\footinsA` 1733, 1746, 2150, 2469, 2475
 - `\footinsB` 1734, 1747, 2437, 2470, 2476
 - `\footinsC` 1735, 1748, 2451, 2471, 2477
 - `\footnormal`
1384, 1406–1410, 1710, 3535, 3536
 - `\footnormalX` 26, 2240, 2462–2464, 2481
 - `\footnote` 3456, 3466, 3545
 - `\footnoteA` 25, 2144, 2513
 - `\footnoteB` . 25, 2428, 2514, 3467–3469
 - `\footnoteC` 25, 2446, 2515
 - `\footnoterule` . 1375, 1660, 2211, 2553
 - `\footnotesize` 1166, 2018,
2019, 4030, 4031, 4167, 4168, 4353
 - `\footparagraph` 16, 1412, 3533, 3656,
3792, 4027, 4028, 4247–4249, 4356
 - `\footparagraphX` 26, 2352, 3429
 - `\footplitskips` 1263,
1266, 1441, 1455, 1554, 1593,
2184, 2280, 2326, 2384, 2397, 4184
 - `\footstartA` 2469
 - `\footstartB` 2470
 - `\footstartC` 2471
 - `\footthreecol` 16, 1534, 3532
 - `\footthreecolX` 26, 2307
 - `\foottwocol` 16, 1578, 3531, 3793
 - `\foottwocolX` 26, 2261
 - `\foottwocolX` 2261
 - `\fullstop`
18, 304, 1289, 1362, 1364, 1366,
1368, 1954, 1958, 3778, 3787, 4434
- ## G
- `\g@addto@macro` 1702, 1703,
1711, 1714, 1717, 1719, 1727,
2482, 2487, 2490, 2493, 2608, 2644
 - Gädeke, Nora 8
 - `\get@linelistfile` 348, 364
 - `\getline@num` 836, 864
 - `\gfootnote` 3924, 4060
 - `\ggfootfmt` 3643, 3657
 - `\ggfootstart` 3660, 3661
 - `\gl@p` 326, 355, 356, 675, 691, 717, 921,
922, 1117, 1121, 1157, 1798, 4425
 - `\gl@poff` 326, 327
 - `\goodbreak` 3789
- ## H
- `\hangafter` 2749, 3498, 3664, 3749
 - `\hangingsymbol` 21, 2698, 2702
 - `\hb@xt@` 839,
841, 857, 859, 3204, 3209, 3213,
3218, 3229, 3242, 3255, 3332, 3334
 - `\hfilneg` 1523
 - `\Hilfsbox` 2809
 - `\hilfsbox` 2809, 2864, 2865,
2907, 2919, 3009, 3022, 3039,
3053, 3067, 3080, 3095, 3220,
3222, 3227, 3231, 3232, 3234,
3240, 3244, 3245, 3247, 3253, 3256
 - `\hilfscount` 2809, 3370–3372, 3378
 - `\HILFSskip` 3363
 - `\Hilfsskip` 3210,
3214, 3215, 3219, 3222, 3223,
3230, 3231, 3234–3236, 3243,
3244, 3247–3249, 3256–3258,
3363, 3369, 3371, 3377, 3381, 3382
 - `\hilfsskip`
. 2809, 3008, 3009, 3024, 3041,
3055, 3069, 3082, 3097, 3380–3382
 - `\hphantom` 2784
 - `\Hy@temp@A` 2632, 2633
 - `\HyInd@ParenLeft` 2633
 - `\hyphenation` 4260
 - `\hyphenpenalty` 3626
- ## I
- `\if@fcolmade` 1766, 1775

- \if@firstcolumn 998, 1073, 1768, 2077
 - \if@led@nofoot 1723, 1753
 - \if@nobreak 761
 - \ifautopar 778, 806
 - \ifbypage@ 183, 423, 434, 900, 1325, 3633
 - \ifbypstart@ 183, 755, 1350
 - \ifdim 611, 613, 615, 617, 2102, 2864, 3370
 - \iffirst@linenum@out@ 589, 593
 - \iffirstpstart 751, 755
 - \ifhbox 1506, 1511
 - \ifhmode 2116, 2123, 2165, 2172
 - \ifinstanza 778, 823, 2698, 2702
 - \ifl@d@dash 1293, 1365, 1955
 - \ifl@d@elin 1293,
1343, 1367, 1368, 1945, 1957, 1958
 - \ifl@d@esl 1293, 1368, 1958
 - \ifl@d@pnum
1293, 1331, 1362, 1366, 1933, 1956
 - \ifl@d@ssub 1293, 1364, 1954
 - \ifl@dend@ 1885, 1891
 - \ifl@dmemoir 28, 2875
 - \ifl@dpairing 114, 151, 1351
 - \ifl@dskipnumber 637, 959
 - \ifl@dstartendok 3174, 3184
 - \ifledfinal 5, 30
 - \ifledplinum 1303, 1363, 4229
 - \ifledRcol 114, 809, 1352
 - \ifleftnoteup 2054, 2068
 - \ifnolinenum 3629, 3648
 - \ifnoteschanged@ 161, 344
 - \ifnumberedpar@ 742, 771,
783, 1174, 1182, 1190, 1198,
1206, 1221, 1229, 1237, 1245,
1253, 1898, 1899, 1902, 1903,
1906, 1907, 1910, 1911, 1914,
1915, 1973, 2029, 2035, 2041,
2129, 2135, 2223, 2234, 3916, 3925
 - \ifnumbering 113,
122, 148, 171, 186, 767, 780, 815
 - \ifnumberingR 114, 810
 - \ifnumberline 697, 865, 870, 958
 - \ifnumberpstart ... 750, 778, 795, 823
 - \ifodd 1008, 1083, 2087, 4003
 - \ifpst@rtedL 114
 - \ifpstartnum 1094, 1097, 1102
 - \ifreportnoidxfile 2614
 - \ifrightnoteup 2026, 2063
 - \ifshowindexmark 2627, 2660
 - \ifsidepstartnum 778, 1068
 - \ifsublines@
. 302, 331, 412, 447, 452, 458,
473, 491, 500, 514, 536, 583,
585, 871, 907, 962, 1817, 1842, 3945
 - \ifvbox 789, 1630, 1698
 - \ifvmode 1799
 - \ifvoid 1650,
1677–1681, 1688–1692, 1696,
1712, 1715, 1720, 1728–1735,
1740–1748, 2469–2471, 2475–
2477, 2485, 2488, 2494, 2506–
2510, 2517–2519, 2541, 2563, 2590
 - \IMM 4393, 4394
 - \indexentry 2653
 - \initnumbering@reg 121
 - \InputIfFileExists 365
 - \insert 1261,
1438, 1552, 1591, 1688–1692,
1696, 1698, 1716, 2182, 2278,
2324, 2381, 2475–2477, 2489, 4181
 - \insert@count 559, 560,
605, 669, 685, 1177, 1185,
1193, 1201, 1209, 1224, 1232,
1240, 1248, 1256, 2032, 2038,
2044, 2138, 2226, 3919, 3928, 4419
 - \inserthangingsymbol 839, 2702
 - \inserthangingymbol 2701
 - \insertlines@list
... 154, 335, 361, 567, 1113, 1117
 - \insertparafootftmsep 1480, 1514, 2407
 - \inserts@list 775, 1108,
1111, 1121, 1156, 1157, 1176,
1184, 1192, 1200, 1208, 1223,
1231, 1239, 1247, 1255, 2031,
2037, 2043, 2137, 2225, 3918, 3927
 - \instanzafalse 2700, 2766
 - \instanzatrue 2755
 - \interfootnotelinepenalty 1267
 - \interlinepenalty 803, 1141, 1267, 2760
 - \interparanoteglue 16, 1474, 3760
 - \ipn@skip 1472, 1474
 - \itshape 3665, 3774, 3780, 3788
- J**
- Jayaditya 8
- K**
- Kabelschacht, Alois 90
 - Krukov, Alexej 67

L

- `\l@d@@wrindexhyp` 2625, 2658
- `\l@d@add` 732, 734, 738, 740
- `\l@d@dashfalse` 1302, 1324, 1928
- `\l@d@dashtrue`
1328, 1334, 1346, 1931, 1936, 1948
- `\l@d@elinfalse` 1298, 1331, 1933
- `\l@d@elintrue` .. 1331, 1333, 1933, 1935
- `\l@d@end` .. 1885, 1887, 1888, 1894,
1897, 1901, 1905, 1909, 1913, 1970
- `\l@d@err@UnequalColumns` 2950
- `\l@d@eslfalse`
.... 1300, 1340, 1343, 1942, 1945
- `\l@d@esltrue` ... 1343, 1345, 1945, 1947
- `\l@d@index` 2610, 2612, 2877
- `\l@d@makecol` 1642, 1705, 1784
- `\l@d@nums` 668, 715, 718, 726, 727, 740,
1176, 1184, 1192, 1200, 1208,
1223, 1231, 1239, 1247, 1255,
1898, 1902, 1906, 1910, 1914, 3917
- `\l@d@pnumfalse` 1294, 1324, 1928
- `\l@d@pnumtrue` 1327, 1930
- `\l@d@reinserts` 1695, 1706
- `\l@d@section` 1894, 1917
- `\l@d@set` 480, 631
- `\l@d@ssubfalse` 1296, 1336, 1938
- `\l@d@ssubtrue` 1338, 1940
- `\l@d@wrindexm@m` 2624, 2625
- `\l@d@ampcount` 2804,
2946, 2948, 2953, 3218, 3228,
3229, 3241, 3242, 3277, 3293, 3331
- `\l@d@begin@stack` 2676, 2686–2688
- `\l@d@bfnote` 2130, 2134
- `\l@d@checkcols` 2903, 2915, 2943
- `\l@d@checkstartend` 3173, 3184
- `\l@d@chset@num` 379, 382, 480
- `\l@d@colcount` 2804, 2846,
2858, 2859, 2902, 2904, 2914,
2916, 2944, 2948, 2953, 3014,
3016, 3031, 3033, 3047, 3048,
3061, 3062, 3074, 3075, 3089,
3090, 3140, 3141, 3218, 3228,
3229, 3241, 3242, 3273, 3275,
3289, 3291, 3331, 3337, 3367, 3376
- `\l@d@collect@body` 2678, 2685
- `\l@d@collect@body`
.... 2673, 3403–3405, 3407–3409
- `\l@d@colwidth` 2846, 2864, 2865,
3008, 3139, 3204, 3230, 3243,
3332, 3334, 3339, 3348, 3369, 3370
- `\l@d@csnote` 2023, 2026
- `\l@d@csnotetext` 851, 2049,
2070, 2075, 2078, 2080, 2088, 2090
- `\l@d@dodoreinextrafeet` 1683, 1697, 1703
- `\l@d@dofootinsert` 1643, 1648
- `\l@d@dextrafeet` 1665, 1668, 1702
- `\l@d@dedbeginmini` 1717, 2496, 2499
- `\l@d@dedendmini` 1719, 2497, 2499
- `\l@d@emptyd@ta` 833, 851
- `\l@d@end@close` 1887, 1963
- `\l@d@end@false` 1885, 1888
- `\l@d@end@open` 1887, 1892
- `\l@d@end@stuff` ... 131, 176, 1890, 1969
- `\l@d@end@true` 1885, 1887
- `\l@d@envbody` 2668, 2671, 2674–2676
- `\l@d@fambeginmini` ... 2490, 2496, 2512
- `\l@d@famendmini` 2493, 2497, 2512
- `\l@d@feetbeginmini`
..... 2496, 2533, 2559, 2586
- `\l@d@feetendmini` 2496, 2544, 2566, 2593
- `\l@d@getline@margin` 209
- `\l@d@getlock@disp` 253, 281
- `\l@d@getref@num` 1844,
1845, 1847, 1848, 1850, 1851, 1858
- `\l@d@getsidenote@margin` 1980
- `\l@d@gobblearg` 2894, 2975–2979
- `\l@d@gobbledarg` 2894, 2958–2962
- `\l@d@label@parse` 1864, 1867
- `\l@d@d@ta` ... 839, 851, 997, 1074, 1086
- `\l@d@lp@rbox` 857, 2011, 2053, 2059
- `\l@d@lsn@te` 840, 856
- `\l@d@lsnote` 2021, 2026
- `\l@d@make@labels` 1803, 1806, 1828, 1837
- `\l@d@memoirfalse` 29
- `\l@d@memoirtrue` 29
- `\l@d@modforcritext` 2956, 3412
- `\l@d@modfordtext` 2973, 3415
- `\l@d@nullfills` 2990,
3262, 3280, 3296, 3306, 3314, 3324
- `\l@d@numpstartsL` 114, 137
- `\l@d@dold@footnotetext` 2110, 2112
- `\l@d@dold@xympar` 1971
- `\l@d@dold@dold@footnotetext` . 2127, 2142
- `\l@d@p@rsefootspec` 1307
- `\l@d@pairingfalse` 114
- `\l@d@pairingtrue` 114
- `\l@d@parsedendline` 1307, 4191
- `\l@d@parsedendpage` 1307
- `\l@d@parsedendsub` 1307

- \l@dparsedstartline 1307, 4186, 4191, 4193
- \l@dparsedstartpage . 1307, 4189, 4194
- \l@dparsedstartsub 1307
- \l@dparsefootspec 1307, 4185
- \l@dpush@begins 2682, 2686
- \l@drd@eta ... 841, 851, 997, 1076, 1084
- \l@dref@undefined 1844, 1847, 1850, 1853
- \l@drestorefills 2990, 3266, 3282, 3300, 3308, 3318, 3326
- \l@drestoreforcritext ... 2956, 3413
- \l@drestoreforedtext 2973, 3416
- \l@drp@rbox 859, 2011, 2062
- \l@drsn@te 842, 856
- \l@drsnote 2022, 2026
- \l@dsetmaxcolwidth .. 2863, 2909, 2921
- \l@dskipnumberfalse 637, 960
- \l@dskipnumbertrue 637, 951
- \l@dstartendokfalse . 3188, 3192, 3196
- \l@dstartendoktrue 3186
- \l@dtabaddcols 3172, 3203
- \l@dtabnoexpands 660, 2777
- \l@dunboxmpfoot 2542, 2550, 2564, 2591
- \l@dunhbox@line 828
- \l@dzeropenalties 786, 801
- Lück, Uwe 6
- \label 24, 3444
- \label@refs 1796, 1798, 1803, 1806, 1814, 1815, 1819, 1821, 1823
- \labelref@list . 1789, 1795, 1798, 1842
- \labelrefsparseline 1809
- \labelrefsparsesubline 1809
- \last@page@num 429
- \lastbox ... 821, 835, 1467, 1505, 1510
- \lastkern 2102
- \lastskip 610, 614
- Lavagnino, John 4, 6
- \lbreak 4394, 4458
- \ldots 4294, 4296, 4322
- Leal, Jeronimo@Leal, Jerónimo 6
- \led@err@AutoparNotNumbered 72, 811, 816
- \led@err@HighEndColumn 99, 3193
- \led@err@LineationInNumbered 49, 187
- \led@err@LowStartColumn ... 99, 3189
- \led@err@NumberingNotStarted 33, 165
- \led@err@NumberingShouldHaveStarted 33, 178
- \led@err@NumberingStarted ... 33, 123
- \led@err@PendNoPstart 72, 784
- \led@err@PendNotNumbered 72, 781
- \led@err@PstartInPstart 72, 772
- \led@err@PstartNotNumbered .. 72, 768
- \led@err@ReverseColumns ... 99, 3197
- \led@err@TooManyColumns ... 99, 2860
- \led@err@UnequalColumns 99
- \led@mess@NotesChanged 39, 162
- \led@mess@SectionContinued .. 47, 174
- \led@warn@BadAction 86, 953
- \led@warn@BadAdvancelineLine 62, 467
- \led@warn@BadAdvancelineSubline 62, 461
- \led@warn@BadLineation 52, 204
- \led@warn@BadLinenummargin .. 52, 232
- \led@warn@BadLockdisp 52, 259
- \led@warn@BadSetline 68, 622
- \led@warn@BadSetlinenum 68, 629
- \led@warn@BadSidenotemargin 95, 2003
- \led@warn@BadSubblockdisp 52, 285
- \led@warn@DuplicateLabel ... 88, 1831
- \led@warn@NoIndexFile 97, 2615
- \led@warn@NoLineFile 60, 370
- \led@warn@NoMarginpars 93, 1974
- \led@warn@RefUndefined 88, 1855
- \ledfinalfalse 7
- \ledfinaltrue 6
- \ledfootinsdim 1384, 1397, 1402, 2251, 2257
- ledgroup (environment) 22, 2556
- ledgroupsize (environment) . 22, 2569
- \ledleftnote 25, 2021
- \ledlinenum 297
- \ledllfill 841, 861, 2573, 2577
- \ledlsnotefontsetup ... 25, 2014, 2052
- \ledlsnotesep 25, 857, 2014
- \ledlsnotewidth 25, 2014, 2052
- \ledmac@error 32, 34, 36, 38, 50, 73, 76, 79, 82, 84, 100, 102, 105, 107, 109, 3198
- \ledmac@warning 31, 53, 55, 57, 59, 61, 63, 66, 69, 71, 87, 89, 91, 94, 96, 98, 1814
- \ledplinenumfalse 31, 4187
- \ledplinenumtrue 31, 1304, 4185, 4190, 4192
- \ledrightnote 25, 2021, 3457
- \ledrlfill 841, 861, 2574, 2581
- \ledrsnotefontsetup ... 25, 2014, 2061
- \ledrsnotesep 25, 859, 2014

- `\ledrsnotewidth` . 25, 2014, 2061, 3433
 - `\ledsetnormalparstuff`
 - 1281, 1481, 2198, 2408,
 - 3487, 3519, 3771, 3777, 4038,
 - 4046, 4218, 4226, 4239, 4361, 4366
 - `\ledsidenote` 25, 2021, 4273
 - `\left` 3147, 3150, 3155, 3158
 - `\leftctab` 3217, 3315
 - `\lefthyphenmin` 3901
 - `\leftlinenum`
 - . . . 12, 297, 999, 1011, 3638, 4007
 - `\leftltab` 3208, 3297
 - `\leftnoteuptrue` 2069
 - `\leftpstartnum` 1068
 - `\leftrtab` 3212, 3263
 - Leibniz 8
 - `\lemma` 14, 724, 3556, 3591,
 - 3714, 3717, 3813, 3824, 3849,
 - 3874, 4071, 4077, 4080, 4095, 4322
 - `\lemmfont` 4041, 4056, 4221, 4241
 - `\letsforverteilen` 2998,
 - 3023, 3040, 3054, 3068, 3081, 3096
 - Levy, Silvio 190
 - `\line` 1813, 1815, 1816, 1821, 1823
 - `\line@list` 157, 335, 360, 585, 713, 717
 - `\line@list@stuff` 130, 175, 591
 - `\line@margin` . . . 209, 1004, 1079, 3999
 - `\line@num` 139, 301, 329, 384,
 - 418, 424, 435, 465, 466, 468,
 - 476, 481, 482, 494, 578, 582,
 - 847, 877, 901, 910, 975, 977,
 - 978, 987, 988, 1841, 2702, 3633–
 - 3635, 3638, 3765, 3972–3974, 4005
 - `\line@set` 728, 729
 - `\lineation` 11, 50, 53,
 - 185, 3527, 3641, 3798, 4025, 4163
 - `\linenum` 15, 725, 1880,
 - 2893, 2964, 2981, 3007, 4408, 4420
 - `\linenum@out` . . . 588, 594, 596, 598,
 - 599, 602, 604, 607, 612, 616,
 - 619, 624, 631, 634, 635, 641, 1794
 - `\linenumberlist` 12, 25, 976, 988
 - `\linenumberstyle` 13, 288, 4473
 - `\linenumincrement` 10, 12, 248
 - `\linenummargin` 11, 55,
 - 209, 3528, 3640, 3799, 4026, 4164
 - `\linenumr@p` 288
 - `\linenumrep` 288, 301,
 - 1363, 1367, 1841, 1953, 1957, 4434
 - `\linenums` 3631
 - `\linenumsep` 12, 297, 1098,
 - 1103, 2016, 2017, 3746, 3757, 3758
 - `\lineref` 23, 655, 1847, 2605, 3471
 - `\linewidth` 839
 - `\list@clear` 315, 360–363, 775
 - `\list@clearing@reg` 347, 359
 - `\list@create`
 - . . . 314, 335–338, 644, 1108, 1789
 - `\lmarpar` 4059, 4078
 - `\lock@disp` 253,
 - 1040, 1044, 1048, 3984, 3988, 3992
 - `\lock@off` 507, 508, 534, 635
 - `\lock@on` 505, 634
 - `\lockdisp` 12, 57, 253
 - Lorch, Richard 7
 - `\lp@rbox` 3913, 3936
 - `\ltab` 2789, 3295, 3403
 - `\ltabtext` 2791, 3305, 3407
 - Luecking, Dan 43
- M**
- `\m@m@makecolfloats` 1625, 1644
 - `\m@m@makecolintro` 1625
 - `\m@m@makecoltext` 1625, 1645
 - `\m@mdodoreinextrafeet` 1703
 - `\m@mdoextrafeet` 1702
 - `\m@mmf@check` 2101, 2118, 2167
 - `\m@mmf@prepare`
 - 2098, 2113, 2122, 2148,
 - 2160, 2171, 2432, 2443, 2450, 2459
 - `\m@th` 3165
 - `\Ma` 3903, 4101, 4128, 4137
 - `\makehboxofhboxes`
 - 1487, 1497, 1502, 2414, 2423
 - `\makeindex` 2608, 2644
 - `\makememindexhook` 2608
 - `\maketitle` 3438, 4262
 - `\managestanza@modulo` 2724, 2741
 - `\marg` 15
 - `\marginparwidth` 2014, 2015
 - `\mathchardef` 2719, 4393
 - `\maxdepth` 1646
 - `\maxdimen` 1442, 1456, 2385, 2398, 4195
 - Mayer, Gyula 8
 - `\measurebody` . . 3265, 3271, 3299, 3317
 - `\measurecell` 2901, 2927
 - `\measuremrow` 2925, 3276
 - `\measuretbody` . . 3281, 3287, 3307, 3325
 - `\measuretcell` 2913, 2932
 - `\measuretrow` 2930, 3292

- \medskip 4074
 - \message 48, 129
 - \mid 4128, 4137
 - Middleton, Thomas 8, 55
 - minipage (environment) 22
 - Mittelbach, Frank 7
 - \morenoexpands . . . 37, 647, 3909, 4213
 - \moveleft 3210, 3215, 3223
 - \moveright 3236, 3249, 3258
 - \mpAfootgroup 2506
 - \mpAfootins 1214, 2506
 - \mpAfootnote 1220, 2500
 - \mpBfootgroup 2507
 - \mpBfootins 1214, 2507
 - \mpBfootnote 1220, 2501
 - \mpCfootgroup 2508
 - \mpCfootins 1214, 2508
 - \mpCfootnote 1220, 2502
 - \mpDfootgroup 2509
 - \mpDfootins 1214, 2509
 - \mpDfootnote 1220, 2503
 - \mpEfootgroup 2510
 - \mpEfootins 1214, 2510
 - \mpEfootnote 1220, 2504
 - \mpfootgroupA 2517
 - \mpfootgroupB 2518
 - \mpfootgroupC 2519
 - \mpfootinsA 2156, 2517
 - \mpfootinsB 2439, 2518
 - \mpfootinsC 2455, 2519
 - \mpfootnoteA 2156, 2513
 - \mpfootnoteB 2439, 2514
 - \mpfootnoteC 2455, 2515
 - \mpnormalfootgroup 1378, 1400
 - \mpnormalfootgroupX 2216, 2254
 - \mpnormalvfootnote 1272, 1399, 1539, 1583
 - \mpnormalvfootnoteX 2188, 2253, 2266, 2312
 - \mppara@footgroup 1420, 1492
 - \mppara@footgroupX 2360, 2412
 - \mppara@vfootnote 1419, 1451
 - \mppara@vfootnoteX 2359, 2380
 - \mpthreecolfootgroup 1540, 1570
 - \mpthreecolfootgroupX 2313, 2340
 - \mpthreecolfootsetup 1541, 1547
 - \mpthreecolfootsetupX 2314, 2316
 - \mptwocolfootgroup 1584, 1610
 - \mptwocolfootgroupX 2267, 2295
 - \mptwocolfootsetup 1585, 1610
 - \mptwocolfootsetupX 2268, 2270
 - \mpvAfootnote 1222, 1226
 - \mpvBfootnote 1230, 1234
 - \mpvCfootnote 1238, 1242
 - \mpvDfootnote 1246, 1250
 - \mpvEfootnote 1254, 1258
 - \mpvfootnoteA 2160
 - \mpvfootnoteB 2443
 - \mpvfootnoteC 2459
 - \multfootsep 25, 2095, 2105
 - \multiplefootnotemarker 2095, 2099, 2100, 2102
- N
- \n@num 555, 641
 - \n@num@reg 555
 - \NeedsTeXFormat 2
 - \new@line 602, 841
 - \newbox 742, 745, 2011, 2012, 2809, 2811, 3400, 3401, 3913
 - \newcounter 239, 241, 243, 245, 748, 884, 2151, 2434, 2451, 2598, 2727, 3170, 4397
 - \newif 5, 28, 113, 114, 116, 119, 120, 183, 184, 331, 344, 589, 637, 697, 743, 750, 751, 806, 1069, 1094, 1293, 1295, 1297, 1299, 1301, 1303, 1723, 1886, 2026, 2068, 2699, 3184, 3629
 - \newinsert . . . 1214–1218, 1386–1388, 2150, 2161, 2437, 2444, 2453, 2460
 - \newlength 297, 2711
 - \newlinechar 1896, 1900, 1904, 1908, 1912
 - \newparafootfmt 3776, 3794, 3827
 - \newread 345
 - \newtwocolfootfmt 3782, 3795
 - \newwrite 588, 1885
 - \NEXT 2897, 2902, 2905, 2910, 2911, 2914, 2917, 2922, 2923, 2926, 2928, 2929, 2931, 2933, 2934, 2939, 3102, 3105, 3107, 3108, 3110, 3112, 3113, 3116, 3118, 3119, 3121, 3123, 3124, 3127, 3129, 3130, 3132, 3134, 3135, 3140, 3142, 3143, 3337, 3340, 3341, 3346, 3350, 3351, 3367, 3373, 3374
 - \Next 2939, 3015, 3017, 3026, 3027, 3032, 3034,

- 3043, 3044, 3047, 3049, 3056,
 3057, 3061, 3063, 3070, 3071,
 3074, 3076, 3084, 3085, 3089,
 3091, 3099, 3100, 3355, 3357, 3358
 \next@action 87, 356,
 890, 898, 899, 904, 905, 913, 922
 \next@actionline
 353, 355, 889, 897, 919, 921
 \next@insert
 776, 1112, 1115, 1117, 1120, 1124
 \next@page@num
 144, 387, 389, 427, 439, 488
 \no@expands .. 647, 666, 682, 3907, 4416
 \noalign 1526
 \nobrak 4211, 4212
 \noendnotes 23,
 1969, 3431, 3481, 3621, 3755, 3899
 \noindent 823, 1066, 1443,
 1457, 1490, 1500, 2386, 2399,
 2417, 2426, 4070, 4077, 4196, 4269
 \nolinenums 3628, 3630
 \nolinenumsfalse 3631
 \nolinenumstrue 3630
 \nonumparafootfmt .. 3769, 3770, 3808
 \normal@footnotemarkX ... 2174, 2242
 \normal@pars
 ... 150, 777, 826, 1282, 1557,
 1596, 2285, 2330, 3493, 3507, 3783
 \normalbfnoteX 2222, 2235
 \normalbodyfootmarkX 2179, 2243
 \normalcolor 1380, 1494, 1572, 1615,
 1659, 2218, 2301, 2346, 2420, 2552
 \normalfont 299, 1167, 2096, 2180, 3789
 \normalfootfmt . 1281, 1392, 3486, 3534
 \normalfootfmtX 2197, 2246
 \normalfootfootmarkX 2202, 2247
 \normalfootgroup 1376, 1393
 \normalfootgroupX 2213, 2248
 \normalfootnoterule 1375, 1395
 \normalfootnoteruleX 2211, 2249, 2357
 \normalfootstart 1371, 1390
 \normalfootstartX 2205, 2241
 \normalvfootnote 1260, 1391
 \normalvfootnoteX 2181, 2244
 \nospeak 3675, 3686
 \nospeaker 3674
 \note 3797, 3838, 3853, 3858, 3864, 3875
 \notefontsetup 17, 1166,
 1262, 1275, 1425, 1440, 1454,
 1476, 1489, 1499, 1553, 1566,
 1592, 1605, 1917, 2183, 2191,
 2279, 2295, 2325, 2340, 2364,
 2383, 2396, 2416, 2425, 3644, 4183
 \notenumfont 17,
 1167, 1286, 1482, 1563, 1602,
 1917, 2199, 2292, 2337, 2409,
 3488, 3500, 3513, 3520, 3526,
 3778, 3787, 4030, 4040, 4048,
 4167, 4220, 4228, 4240, 4362, 4367
 \noteschanged@false 344, 366
 \noteschanged@true
 155, 158, 344, 371, 714, 1114
 \notetextfont 4031, 4042, 4050, 4168,
 4222, 4235, 4242, 4353, 4364, 4369
 \nulledindex 2874, 2963, 2980,
 3013, 3030, 3046, 3060, 3073, 3088
 \nullsetzen 3137, 3274, 3290
 \num@lines . 742, 787, 1131, 1137, 1140
 \numberedpar@false 742
 \numberedpar@true 742, 779
 \numberingfalse 113, 149
 \numberingtrue 113, 126, 169
 \numberit 4399, 4404
 \numberlinefalse 11
 \numberlinetrue 11, 698
 \numberpstartfalse 11, 747, 753
 \numberpstarttrue 11, 747
 \numlabfont .. 18, 297, 3635, 3638, 3765
- O**
- \oldprintlines 4436
 \one@line 742, 834, 835, 841
 \openout 596, 599, 1887
 \os 3624, 3649, 3718, 3728
 \overfullrule 3900
- P**
- \PackageError 32
 \PackageWarning 31
 \page@action 388, 486, 572
 \page@num 340,
 351, 426, 437, 577, 582, 899,
 1006, 1081, 1516, 1519, 2085, 4001
 \page@start 608, 1649
 \pagelinesep 27, 2596, 2605
 \pageno 89, 91, 1621
 \pageparbreak 37, 1066
 \pageref 24
 \pagestyle 3805, 4441

- `\par@line` 742, 788, 1132, 1133, 1136, 1140
 - `\para@footgroup` 1416, 1485
 - `\para@footgroupX` 2356, 2412
 - `\para@footsetup` 1418, 1425
 - `\para@footsetupX` 2362, 2364
 - `\para@vfootnote` 1414, 1437, 4180
 - `\para@vfootnoteX` 2354, 2380
 - `\parafootfmt` 1415, 1479, 3518
 - `\parafootfmtX` 2355, 2406
 - `\parafootftmsep` 1513, 1517
 - `\parafootstart` 1413, 1433
 - `\parafootstartX` 2353, 2372
 - `\pausenumbering` 10, 168
 - `\pend` . 9, 79, 82, 773, 780, 824, 1066, 2764, 2766, 3460, 3576, 3611, 3666, 3674, 3693, 3706, 3734, 3818, 3825, 3846, 3881, 3888, 4073, 4132, 4146, 4447, 4459, 4464
 - `\phantom` 2783
 - Pigman, IIIrd, G. W. 184
 - Plato of Tivoli 7
 - `\postbodyfootmark` 2163, 2177
 - `\postdisplaypenalty` 804
 - `\prebodyfootmark` 2163, 2175
 - `\predisdisplaypenalty` 803
 - `\prevgraf` 787
 - `\previous@A@number` 4205
 - `\previous@B@number` 4206
 - `\previous@C@number` 4207
 - `\previous@page` 4189, 4194, 4208
 - `\prevline@num` 847, 2701
 - `\prevpage@num` 1514
 - `\printendlines` 1917, 1951
 - `\printlines` 1286, 1349, 1482, 1563, 1602, 3488, 3500, 3513, 3520, 3649, 3778, 3787, 4040, 4048, 4220, 4228, 4240, 4362, 4367, 4436, 4471
 - `\printnpnum` 22, 1953, 1956, 1961
 - `\printstanzalines` 4433, 4471
 - `\processl@denvbody` 2675, 2679, 2680, 2695
 - `\ProcessOptions` 9
 - `\protected@write` 1805, 2629, 2634, 2637, 2652, 2661
 - `\protected@xdef` 2146, 2158, 2430, 2441, 2448, 2457
 - `\ProvidesPackage` 3
 - `\pst@rtedLfalse` 114, 136, 152
 - `\pst@rtedLtrue` 114, 172
 - `\pstart` 9, 73, 76, 77, 82, 747, 823, 1066, 2745, 3447, 3542, 3578, 3663, 3668, 3672, 3674, 3675, 3749, 3811, 3821, 4068, 4076, 4134, 4447, 4455, 4461
 - `\pstartnum` 1068
 - `\pstartnumfalse` 1099, 1106
 - `\pstartnumtrue` 797, 1095
- Q**
- `\quad` 3861, 3878
- R**
- `\raggedright` 1561, 1600, 2019, 2290, 2335, 3497, 3511
 - `\raw@text` 742, 778, 789, 834
 - `\rbracket` ... 18, 1287, 1289, 1483, 1564, 1603, 3501, 3773, 3779, 3788, 4049, 4211, 4234, 4363, 4368
 - `\rd@ta` 3940, 4005
 - `\read@linelist` 345, 592
 - `\ref` 24, 3472
 - `\refstepcounter` 4403
 - `\Relax` 2897, 3354, 3361
 - `\rem@inder` 988, 990–992
 - `\removehboxes` 1488, 1498, 1502, 2415, 2424
 - `\removelastskip` 3014, 3031
 - `\resumenumbering` 10, 168
 - `\right` 3148, 3151, 3156, 3159
 - `\rightctab` 3226, 3316
 - `\rightlinenum` 12, 297, 1001, 1009, 3633, 3765, 4004
 - `\rightltab` 3239, 3298
 - `\rightnoteuptrue` 2027
 - `\rightpstartnum` 1076, 1084, 1101
 - `\rightrtab` 3252, 3264
 - `\rightstartnum` 1068
 - `\rigidbalance` ... 1521, 1569, 1576, 1608, 1619, 2298, 2305, 2343, 2350
 - `\rlap` 1001, 1009, 1076, 1084, 3746, 4004
 - `\rmarp` 4060, 4069, 4110, 4138
 - `\rmfamily` 3485, 3505, 3652
 - Robinson, Peter 5
 - `\roman` 3171
 - `\rtab` 2787, 3261, 3405
 - `\rtabtext` 2790, 3279, 3409

S

- Sacrobosco 8
 \savel@dcnote 2070
 \sc@n@list 989, 991
 \scf 3762,
 3859, 3864, 3867, 3868, 3875, 3876
 Schöpf, Rainer 7
 \section@num
 110, 127, 129, 130, 173–175, 1894
 \select@lemmafонт 649, 1168
 \select@lemmafонт 18,
 1168, 1287, 1483, 1564, 1603,
 1918, 3489, 3501, 3514, 3521,
 3773, 3779, 4049, 4234, 4363, 4368
 \sen 3667,
 3709, 3710, 3712, 3714, 3719–3724
 \senspeak 3668, 3708
 \set@line 668, 684, 701, 712, 4418
 \set@line@action 381, 471, 478, 489, 574
 \setl@dlp@rbox . 2047, 2051, 2078, 2090
 \setl@drp@rbox . 2048, 2060, 2080, 2088
 \setl@drpr@box 2051
 \setline ... 13, 69, 620, 657, 758, 3829
 \setlinenum 13, 71, 627, 4405
 \setlp@rbox 3933, 3938
 \setmcellcenter 3072, 3128
 \setmcellleft 3045, 3117
 \setmcellright 3012, 3106
 \setmrowcenter 3126, 3320
 \setmrowleft 3115, 3302
 \setmrowright 3104, 3268
 \setprintendlines 1927, 1952
 \setprintlines 1323, 1361
 \setstanzaindents 20, 2724, 4381
 \setstanzapenalties
 20, 2724, 4386, 4387
 \setstanzavalues 2714, 2724, 2725, 2775
 \settcclcenter 3087, 3133
 \settcclleft 3059, 3122
 \settcclright 3029, 3111
 \settrwcenter 3131, 3328
 \settrwleft 3120, 3310
 \settrwright 3109, 3284
 \Setzen 3345, 3354, 3356
 Shakespeare, William 187
 \showlemma . 14, 15, 20, 22, 31, 673, 689
 \sidenote@margin 1980, 2083
 \sidenotemargin 25, 1980, 4165
 \sidenotemmargin 96
 \skip 1372, 1379, 1398,
 1403, 1435, 1493, 1571, 1614,
 1657, 2206, 2217, 2252, 2258,
 2300, 2345, 2373, 2377, 2419,
 2551, 3659, 3660, 4174–4176, 4372
 \skip@lockoff 508, 534
 \skipnumbering 13, 637
 \skipnumbering@reg 637
 \sl 647
 \spacedcolon 3485, 3489, 3521
 \spacefactor
 2103, 2106, 2117, 2123, 2166, 2172
 \spaceskip 1264, 2185, 2281
 \speak 3672, 3695
 \speaker 3749, 3831, 3848, 3883
 \splitmaxdepth 1269
 \splitoff 1521
 \splittopskip 831, 1269, 1523,
 1567, 1569, 1574, 1576, 1606,
 1608, 1617, 1619, 2296, 2298,
 2303, 2305, 2341, 2343, 2348, 2350
 \spreadmath 28, 3333
 \spreadtext 28, 3331
 \ss 2778
 \stage 3663, 3684, 3746, 3752
 \stanza 20, 2753, 4478, 4481, 4498, 4512
 \stanza@count 2705, 2716,
 2719, 2721, 2738, 2750, 2757, 2765
 \stanza@hang 2736, 2759
 \stanza@line 2736, 2765, 2767
 \stanza@modulo
 2728, 2731–2733, 2742, 2757
 \stanzaindentbase 20, 2705,
 2739, 2743, 2748, 2769, 4376, 4400
 \startlock 12, 634, 652, 2746
 \startstanzahook 21, 2753, 4403
 \startsub 12, 610, 651, 3663
 \stepcounter 2145, 2157,
 2429, 2440, 2447, 2456, 2601, 3179
 \stepl@dcclcount 2858, 2908,
 2920, 3021, 3038, 3052, 3066,
 3079, 3094, 3138, 3338, 3347, 3368
 \strip@pt 1431, 2370
 \strip@szacnt 2714
 \sub@action 397, 498, 573
 \sub@change 145,
 391, 392, 398, 448, 450, 453, 455
 \sub@lock 142, 333,
 406, 408, 410, 413, 516, 517,
 519, 520, 538, 539, 541, 872,

943, 945, 946, 948, 1023, 1059,
1061, 1063, 3957, 4018, 4020, 4022
`\sub@off` 447, 616
`\sub@on` 447, 612
`\subline` 1818–1821
`\subline@num` 140, 303,
304, 330, 414, 418, 424, 435,
459, 460, 462, 474, 492, 579,
583, 873, 878, 901, 908, 963–
965, 1842, 3636, 3639, 3946–3948
`\sublinenumberstyle` 13, 288
`\sublinenumincrement` 10, 12, 248
`\sublinenumr@p` 288
`\sublinenumrep` 288,
304, 1364, 1368, 1842, 1954, 1958
`\sublineref` 23, 656, 1850
`\sublines@false` ... 143, 331, 395, 933
`\sublines@true` 331, 393, 931
`\sublock@disp` 279,
1025, 1029, 1033, 3959, 3963, 3967
`\sublockdisp` 59, 279
`\subsection` 3444
Sullivan, Wayne 7, 8, 20,
36, 43, 48, 102, 103, 118, 149, 190
`\symplinenum` 31, 1303, 1363
`\sza@penalty` 2736, 2763, 2764

T

`\tabellzwischen` 3336, 3344
`\tabelskip` 3348, 3388–3390, 3396–3398
`\tabHilfbox` 3387,
3389, 3391, 3395, 3397, 3399, 3400
`\tabhilfbox` 3386,
3388, 3390, 3394, 3396, 3398, 3400
`\tableofcontents` 3439
Tapp, Christian 6
`\textbf` 4095, 4400, 4447, 4457
`\textheight` 1771, 4350
`\textnormal` 1289–1291,
3814, 3832, 3851, 3861, 3862,
3870, 3872, 3874, 3878, 3887, 4212
`\textrm` 3667, 3668, 3859
`\textsc` 4273
`\textsf` 3441, 3448
`\textsuperscript`
.... 2096, 2154, 2180, 2203, 4096
`\textwidth` 2529, 2571, 4351
`\thanks` 3435
`\theadcolcount` 3170, 3177, 3180
`\thefootnoteA` 26, 2146, 2151, 2154, 2158

`\thefootnoteB` .. 2430, 2434, 2441, 3428
`\thefootnoteC` 2448, 2451, 2457
`\thelabidx` 2602, 2605
`\thempfn` 2531, 2557, 2584
`\thempfootnote` 2531, 2557, 2584
Theodosius 8
`\thepage` 602, 1806, 2605
`\thepageline`
2604, 2630, 2635, 2638, 2653, 2662
`\thepstart` 11,
747, 749, 778, 823, 1097, 1104, 1358
`\thepstartL` 1355
`\thepstartR` 1353
`\thestanzanum` 4400
`\thinspace` 1291, 3485, 3505
`\thr@@` 230, 519, 528, 539,
546, 938, 946, 1546, 1549, 1569,
1576, 2001, 2318, 2321, 2343, 2350
`\threecolfootfmt` ... 1536, 1556, 3492
`\threecolfootfmtX` 2309, 2329
`\threecolfootgroup` 1537, 1566
`\threecolfootgroupX` 2310, 2340
`\threecolfootsetup` 1538, 1544
`\threecolfootsetupX` 2311, 2316
`\threecolvfootnote` 1535, 1551
`\threecolvfootnoteX` 2308, 2323
`\tiny` 3762
`\title` 3434, 4258
`\tolerance` 1560,
1599, 2289, 2334, 3496, 3510, 3785
`\twocolfootfmt` 1580, 1588, 3506
`\twocolfootfmtX` 2263, 2284
`\twocolfootgroup` 1581, 1588
`\twocolfootgroupX` 2264, 2295
`\twocolfootsetup` 1582, 1588
`\twocolfootsetupX` 2265, 2270
`\twocolvfootnote` 1579, 1588
`\twocolvfootnoteX` 2262, 2277

U

`\underbrace` 2785
`\unhbox` 828, 1468, 1488, 1490, 1498,
1500, 1507, 1511, 2415, 2417,
2424, 2426, 3390, 3391, 3398, 3399
`\unkern` 2104
`\unpenalty` 1471, 1504
`\unvbox` 835,
1274, 1376, 1382, 1453, 1466,
1486, 1496, 1532, 1636, 1656,
1661, 1674, 1688–1692, 1696,

- 1698, 1716, 1754, 2190, 2214,
2220, 2395, 2413, 2422, 2468,
2475–2477, 2484, 2489, 2548, 2554
`\unvvh` 1444, 1458, 1465, 2387, 2400, 4197
`\usepackage` 3423, 3479, 3619,
3743, 3897, 3905, 4160, 4161, 4348
`\usingcritext` 3411
`\usingdtext` 3411
- V**
- `\vAfootnote` 1175, 1179
`\valign` 1524
`\value` 2732, 2737, 3176
Vamana 8
`\variab` 2941, 3267,
3283, 3301, 3309, 3319, 3327, 3360
`\vbadness` 830, 1523
`\vbfnoteX` 2224, 2229
`\vBfootnote` 1183, 1187
`\vbox` 778, 1273,
1442, 1452, 1456, 1466, 1633,
1653, 1673, 1779, 1783, 2055,
2057, 2059, 2064, 2066, 2189,
2385, 2394, 2398, 2467, 2483,
2526, 3147, 3150, 3155, 3158,
3162, 3164, 3165, 3209, 3213,
3218, 3229, 3242, 3255, 3936, 4195
`\vCfootnote` 1191, 1195
`\vDfootnote` 1199, 1203
`\vEfootnote` 1207, 1211
`\vffootnote` 3917, 3921, 3938
`\vfil` 1524, 1783,
3148, 3151, 3156, 3159, 3162, 3165
`\vfootnoteA` 2148
`\vfootnoteB` 2432
`\vfootnoteC` 2450
`\vgfootnote` 3926, 3930, 3940
`\vl@dbfnote` 2134
`\vl@dcsnote` 2042, 2047
`\vl@dlsnote` 2030, 2047
`\vl@drsnote` 2036, 2047
- `\vnumfootnoteX` 2233, 2245
`\vrule` 3147, 3150, 3155, 3158, 3162
`\vsize` 1384, 2256, 3658
`\vsplit` 834, 1531, 1754
- W**
- `\wd` 823, 841, 1446, 1460, 2389,
2402, 2864, 2865, 3009, 3222,
3231, 3234, 3244, 3247, 3256,
3388, 3389, 3396, 3397, 3670, 4199
Whitney, Ron 7
`\widowpenalty` 804, 1138
Wujastyk, Dominik 4, 6
- X**
- `\x@lemma` 675–677, 691–693, 4425–4427
`\xcritext` 2868, 2999
`\xedindex` 2874, 2971, 2988, 3001
`\xedlabel` 2872, 3010
`\xedtext` 2868, 3000
`\xleft@appenditem` 322
`\xlineref` 23, 1847
`\xpageref` 23, 1844
`\xparafootfmt` 4360, 4371, 4469
`\xright@appenditem` 316, 487,
488, 490, 497, 499, 501, 503,
513, 515, 524, 535, 537, 544,
557, 558, 567, 581, 1175, 1183,
1191, 1199, 1207, 1222, 1230,
1238, 1246, 1254, 1840, 2030,
2036, 2042, 2136, 2224, 3917, 3926
`\xspaceskip` 1264, 2185, 2281
`\xsublineref` 23, 1850
`\xxref` 24, 1875
- Y**
- `\yparafootfmt` 4365, 4453
- Z**
- `\z@skip` 1264, 1270, 2185, 2281
`\zz@@@` 1790, 1796, 1877, 1879

Change History

v0.1	General: First public release	1		
v0.10	General: Corrections to <code>\section</code> and other titles in numbered sections	1		
v0.11	General: Makes it possible to add a symbol on each verse's hang- ing, as in French typogra- phy. Redefines the command <code>\hangingsymbol</code> to define the character.	1		
v0.12	General: For compatibilty with led- par, possibility to use <code>\autopar</code> on the right side.	1		
	Possibility to number the pstart with the commands <code>\numberpstarttrue</code>	1		
	Possibility to number <code>\pstart</code>	11		
	<code>\ifledRcol</code> : Added <code>\ifledRcol</code> and <code>\ifnumberingR</code> for/from ledpar	47		
v0.12.1	General: Don't number <code>\pstarts</code> of stanza.	1		
	The numbering of <code>\pstarts</code> restarts on each <code>\beginnumbering</code>	1		
v0.13	General: New <code>stanzaindentsrepeti-</code> tion counter to repeat stanza in- dents every n verses.	1, 20		
	<code>\managestanza@modulo</code> : New stan- zaindentsrepetition counter to repeat stanza indents every n verses.	150		
v0.13.1	General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes con- flicts with memoir class.	1		
v0.14	General: Tweaked <code>\edlabel</code> to get correct line number if the com-			
	mand is first element of a para- graph.	1		
	<code>\edlabel</code> : Tweaked <code>\edlabel</code> to get correct line number if the command is first element of a paragraph.	118		
v0.15	General: Line numbering can be re- set at each pstart.	49		
	New management of hang- ingsymbol insertion, preventing undesirable insertions.	149		
	Possibility to print <code>\pstart</code> num- ber in side.	11		
	<code>\affixline@num</code> : Line numbering can be disabled.	84		
	<code>\printlines</code> : Line numbering can be reset at each pstart.	98		
v0.16	General: <code>\parafootftmsep</code> com- mand is printed between each paragraphed footnote	101		
v0.2	General: Added tabmac code, and extended indexing	1		
	<code>\ifl@dmemoir</code> : Added <code>\ifl@dmemoir</code> for memoir class having been used	43		
	<code>\ledmac@error</code> : Added <code>\ledmac@error</code> and replaced error messages	44		
	<code>\morenoexpands</code> : Added <code>\l@dtabnoexpands</code> to <code>\no@expands</code>	72		
v0.2.1	<code>\@lab</code> : Removed page setting from <code>\@lab</code>	120		
	General: Added text about normal labeling	24		
	Bug fixes and match with mem- patch v1.8	1		
	Major changes to insert code when memoir is loaded	115		
	<code>\doxtrafeet</code> : Renamed <code>\doxtrafeet</code> to <code>\l@ddoxtrafeet</code>	114		
	<code>\edlabel</code> : Tweaked <code>\edlabel</code> to get correct page numbers	118		
	<code>\l@d@makecol</code> : Rewrote <code>\@makecol</code> , calling it <code>\l@d@makecol</code>	113		

<code>\l@ddodoreintrafeet:</code>	Re-named <code>\dodoreintrafeet</code> to <code>\l@ddodoreintrafeet</code> 114	to <code>\leftlinenum</code> and <code>\rightlinenum</code> 53
<code>\l@ddofootinsert:</code>	Renamed <code>\dofootinsert</code> as <code>\l@ddofootinsert</code> 113	<code>\linenumberlist:</code> Added <code>\linenumberlist</code> mechanism . 43
<code>\m@m@makecolintro:</code>	Added <code>\m@m@makecolfloats</code> , <code>\m@m@makecoltext</code> and <code>\m@m@makecolintro</code> . . . 113	<code>\printendlines:</code> Added <code>\linenumr@p</code> and <code>\sublinenumr@p</code> to <code>\printendlines</code> 124
<code>\morenoexpands:</code>	Removed some <code>\lets</code> from <code>\no@expands</code> . These were in EDMAC but I feel that they should not have been as they disabled page/line refs in footnotes 72	<code>\printlines:</code> Added <code>\linenumr@p</code> and <code>\sublinenumr@p</code> to <code>\printlines</code> 98
<code>\zz@@@:</code>	Minor change to <code>\zz@@@</code> . 118	<code>\sublinenumr@p:</code> Added <code>\linenumberstyle</code> and <code>\sublinenumberstyle</code> . . 52
v0.2.2	General: Added the Dekker example 196 Improved paragraph footnotes . 1 New Dekker example 1	v0.3.1 General: Not released. Added remarks about the parallel package 1
<code>\footfudgefiddle:</code>	Added <code>\footfudgefiddle</code> 101	v0.31 General: Added remarks about ledmac/parallel package incompatibility 38
<code>\l@d@section:</code>	Used <code>\providecommand</code> for <code>\@gobblethree</code> to avoid clash with the amsfonts package 123	v0.4 <code>\@iiiminipage:</code> Modified kernel <code>\@iiiminipage</code> and <code>\endminipage</code> to cater for critical footnotes 141
<code>\line@list@stuff:</code>	Added initial write of page number in <code>\line@list@stuff</code> 67	General: Added <code>\showlemma</code> to <code>\edtext</code> (and <code>\critext</code>) . . . 74 Added minipage, etc., support . 1
<code>\para@footsetup:</code>	Added <code>\footfudgefiddle</code> to <code>\para@footsetup</code> 101	<code>ledgroup:</code> Added <code>ledgroup</code> environment 142
<code>\para@footsetupX:</code>	Added <code>\footfudgefiddle</code> to <code>\para@footsetupX</code> 137	<code>ledgroupsize:</code> Added <code>ledgroupsize</code> environment 142
<code>\symlinenum:</code>	Added <code>\symlinenum</code> 96	<code>\footnormal:</code> Added minipage footnote setup to <code>\footnormal</code> . 100
v0.3	<code>\@l@reg:</code> Added a bunch of code to <code>\@l</code> for handling <code>\setlinenum</code> 60 <code>\@lab:</code> Replaced <code>\the\line@num</code> by <code>\linenumr@p\line@num</code> in <code>\@lab</code> , and similar for sub-lines 120 General: Added the Braonain example 200 Includes edstanza and more . . . 1 Two more Dekker examples . . 32	<code>\ifledfinal:</code> Added final/draft options 43 <code>\l@dfeetendmini:</code> Added <code>\l@dfeetbeginmini</code> , <code>\l@dfeetendmini</code> and all their supporting code 140 <code>\mpEfootins:</code> Added <code>\mpAfootins</code> and friends 93 <code>\mpEfootnote:</code> Added <code>\mpAfootnote</code> and friends 93 <code>\mpfootinsA:</code> Familiar footnotes extended for minipages . . . 131 <code>\mpfootinsB:</code> Familiar footnotes extended for minipages . . . 139 <code>\mpfootinsC:</code> Familiar footnotes extended for minipages . . . 139
<code>\ledlinenum:</code>	Added <code>\linenumr@p</code> and <code>\sublinenumr@p</code>	

<code>\mpnormalfootgroup:</code>	Added	Extended <code>\@l</code> to include the page number	60
<code>\mpnormalfootgroup</code> 99		
<code>\mpnormalvfootnote:</code>	Added	<code>\@lopR:</code> Added <code>\@pend</code> , <code>\@pendR</code> , <code>\@lopL</code> and <code>\@lopR</code> in anticipation of parallel processing	62
<code>\mpnormalvfootnote</code> 94		
v0.4.1		General: Changed version of the Dekker example	196
<code>\@opxtrafeetii:</code>	Added <code>\@opxtrafeetii</code>	Fixed long paragraphs looping	1
.....	114	Fixed minor typos	1
General: Added code for changing		Prepared for ledpar package	1
<code>\@doclearpage</code> 116		
Let ledmac take advantage of memoir's indexing 144	<code>\fix@page:</code> Added <code>\last@page@num</code> and <code>\fix@page</code>	61
Not released. Minor editorial improvements and code tweaks	.. 1	<code>\footnoteA:</code> Modified <code>\footnoteA</code> and friends to include <code>\@thefnmarkA</code> etc	130
Only change <code>\@footnotetext</code> and <code>\@footnotemark</code> if memoir not used 129	<code>\new@line:</code> Extended <code>\new@line</code> to output page numbers	67
<code>\addfootins:</code> Added <code>\addfootins</code>	115	<code>\page@start:</code> Made <code>\page@start</code> a no-op	67
<code>\addfootinsX:</code> Added minpage setup to <code>\addfootinsX</code> 140	<code>\vl@dbfnote:</code> Changed <code>\l@dbfnote</code> and <code>\vl@dbfnote</code> as originals could give incorrect markers in the footnotes	130
<code>\doxtrafeetii:</code>	Changed		
<code>\doxtrafeetii</code> code for easier extensions 114		
<code>\ledfootinsdim:</code> Added <code>\ledfootinsdim</code> 99		
v0.5		v0.7	
<code>\@footnotetext:</code> Enabled regular <code>\footnote</code> in numbered text	130	<code>\@l@reg:</code> Added <code>\@l@reg</code>	60
<code>\@xympar:</code> Eliminated <code>\marginpar</code> disturbance 125	<code>\@ref@reg:</code> Added <code>\@ref@reg</code>	65
General: Added left and right side notes 125	General: Added bits about ledpar package	38
Added sidenotes, familiar footnotes in numbered text 1	ledmac having been available for 2 years, deleted the commented out original edmac texts	1
v0.5.1		Maïeul Rouquette new maintainer	1
General: Added moveable side note	125	Made macros of all messages	44
Fixed right line numbers killed in v0.5 1	Replaced all <code>\interAfootnotelinepenalty</code> , etc., by just <code>\interfootnotelinepenalty</code>	1
<code>\affixline@num:</code>	Changed	Tidying up for ledpar and ledarab packages	1
<code>\affixline@num</code> to cater for sidenotes 84		
<code>ledgroupsize:</code> Only change <code>\hsize</code> in <code>ledgroupsize</code> environment otherwise page number can be in wrong place 142	<code>\affixline@num:</code> Added skipnummering to <code>\affixline@num</code>	84
<code>\l@dgetsidenote@margin:</code> Added <code>\sidenotemargin</code> and <code>\sidenote@margin</code> 126	<code>\do@actions@fixedcode:</code> Added <code>\do@actions@fixedcode</code>	82
v0.6		<code>\do@actions@next:</code> Added number skipping to <code>\do@actions</code>	81
<code>\@l@reg:</code> Added <code>\fix@page</code> to <code>\@l</code>	60	<code>\do@linehook:</code> Added <code>\do@linehook</code> for use in <code>\do@line</code>	80
		<code>\endnumbering:</code> Changed	
		<code>\endnumbering</code> for ledpar	48

\f@x@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \f@x@l@cks	86	\normalbfnoteX: Removed extraneous space from	
\footsplitskips: Added \footsplitskips for use in many footnote styles	94	\normalbfnoteX	133
\get@linelistfile: Added \get@linelistfile	59	\resumenumbering: Changed \resumenumbering for ledpar	48
\ifledRcol: Added \l@dnumpstartsL, \ifl@dpairing and \ifpst@rted for/from ledpar	47	\setprintendlines: Added \setprintendlines for use by \printendlines	124
\initnumbering@reg: Added \initnumbering@reg	47	\setprintlines: Added \setprintlines for use by \printlines	97
\l@dcsnotetext: Added \l@demptyd@ta	80	\skipnumbering@reg: Added \skipnumbering and supports	69
\l@ddofootinsert: Deleted \page@start from \l@ddofootinsert	113	\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	51
\l@dgetline@margin: Added \l@dgetline@margin	50	\sublinenumr@p: Using \linenumrep instead of \linenumr@p	52
\l@dgetlock@disp: Added \l@dgetlock@disp	51	Using \sublinenumrep instead of \sublinenumr@p	52
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	126	\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	133
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	80		
\l@dunboxmpfoot: Added \l@dunboxmpfoot containing some common code	142	v0.8 General: Bug on endnotes fixed : in a // text, all endnotes will print and be placed at the ends of columns ()	1
\l@dzeropenalties: Added \l@dzeropenalties	78	v0.8.1 General: Bug on \edtext ; \critex ; \lemma fixed : we can now us non switching commands	1
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	53	v0.9 General: No more ledpatch. All patches are now in the main file.	1
\line@list@stuff: Deleted \page@start from \line@list@stuff	67	v0.9.1 General: Fix some bugs linked to in- tegrating ledpatch on the main file.	1
\list@clearing@reg: Added \list@clearing@reg	58		
\n@num@reg: Added \n@num	65		