# Typesetting simple verse with LaTeX

Author: Peter Wilson, Herries Press
Maintainer: Will Robertson
http://github.com/wspr/herries-press/

v2.4c 2024/02/12

**Abstract**

The verse package provides some aids for the typesetting of simple verse.

## Contents

# 1 Introduction

The typesetting of a poem should be really be dependent on the particular poem. Individual problems do not usually admit of a general solution, so this document and code should be used more as a guide towards some solutions rather than providing a ready made solution for any particular piece of verse.

This manual is typeset according to the conventions of the LaTeX docstrip utility which enables the automatic extraction of the LaTeX macro source files [GMS94].

Section 3 describes the usage of the verse package and commented source code is in Section 4. Colour is used to indicate input and output material; a blue background indicates LaTeX input source, and a green background shows the corresponding output one should expect.

The doggerel used as illustrative material has been taken from [Wil01].

# 2 Verses in LaTeX *without* this package

LaTeX provides the `verse` environment which is defined as a particular kind of list. Within the environment you use \\ to end a line and a blank line will end a stanza. For example, here is a single stanza poem:

```
\newcommand{\garden}{
  I used to love my garden \\
  But now my love is dead \\
  For I found a bachelor's button \\
  In black-eyed Susan's bed.
}
```

When this is typeset as a normal LaTeX paragraph (with no paragraph indentation) it looks like:

I used to love my garden
But now my love is dead
For I found a bachelor's button
In black-eyed Susan's bed.

Typesetting it within LaTeX's `verse` environment produces:

> I used to love my garden
> But now my love is dead
> For I found a bachelor's button
> In black-eyed Susan's bed.

The stanza could also be typeset within the `alltt` environment, defined in the standard alltt package, using a normal font and no \\ line endings.

```
\begin{alltt}\normalfont
I used to love my garden
But now my love is dead
For I found a bachelor's button
In black-eyed Susan's bed.
\end{alltt}
```

which produces:

I used to love my garden
But now my love is dead
For I found a bachelor's button
In black-eyed Susan's bed.

The `alltt` environment is like the `verbatim` environment except that you can use LaTeX macros inside it.

In the `verse` environment long lines will be wrapped and indented but in the `alltt` environment there is no indentation.

Some stanzas have certain lines indented, often alternate ones. To typeset stanzas like this you have to add your own spacing. For instance:

```
\begin{verse}
There was an old party of Lyme \\
Who married three wives at one time. \\
\hspace{2em} When asked: 'Why the third?' \\
\hspace{2em} He replied: 'One's absurd, \\
And bigamy, sir, is a crime.'
\end{verse}
```

will be typeset in a verse environment as:

> There was an old party of Lyme
> Who married three wives at one time.
> When asked: 'Why the third?'
> He replied: 'One's absurd,
> And bigamy, sir, is a crime.'

Using the `alltt` environment you can put in the spacing via ordinary spaces. That is, this

```
\begin{alltt}\normalfont
There was an old party of Lyme
Who married three wives at one time.
     When asked: 'Why the third?'
     He replied: 'One's absurd,
And bigamy, sir, is a crime.'
\end{alltt}
```

is typeset as

```
There was an old party of Lyme
Who married three wives at one time.
     When asked: 'Why the third?'
     He replied: 'One's absurd,
And bigamy, sir, is a crime.'
```

More exotically you could use the TeX `\parshape` command:

```
\parshape = 5 0pt \linewidth 0pt \linewidth
            2em \linewidth 2em \linewidth 0pt \linewidth
\noindent There was an old party of Lyme \\
Who married three wives at one time. \\
When asked: ‘Why the third?’ \\
He replied: ‘One’s absurd, \\
And bigamy, sir, is a crime.’ \par
```

which will be typeset as:

There was an old party of Lyme
Who married three wives at one time.
When asked: ‘Why the third?’
He replied: ‘One’s absurd,
And bigamy, sir, is a crime.’

All of this is about as much assistance as standard (La)TeX provides.

# 3 The verse package

The code provided by the verse package is meant to help with some aspects of typesetting poetry but does not, and cannot, provide a comprehensive solution to all the requirements that will arise.

A brief introduction is included to get started quickly; see the examples in Section 3.4 for more context.

## 3.1 Brief introduction

The verse package provides the `verse` environment for typesetting verses, overwriting LaTeX's original definition. Every line in a verse environment must end with `\\`, and every stanza within a verse should end with `\\!` (an empty line afterwards is optional for readability). These requirements allow line numbering to work correctly in all cases. Use the `\poemlines{`⟨*N*⟩`}` command to number every ⟨*N*⟩th line of a poem.

Use the `\poemtitle{`⟨*title*⟩`}` command (just before the `verse` environment) to give each poem a title; commands are provided to adjust the formatting and include the poem into the standard table of contents.

Each stanza within a verse may optionally be surrounded by either an `altverse` or `patverse` environment to effect specific typesetting; `altverse` indents every second line of a stanza, and `patverse` allows arbitrary indentation based on the ⟨*pattern*⟩ given by `\indentpattern{`⟨*pattern*⟩`}`. The command `\flagverse`, placed at the very beginning of a stanza places a 'title'; e.g., for numbering and otherwise labelling stanzas.

## 3.2 Comprehensive documentation

### 3.2.1 Main verse environments

`verse` (*env.*) The `verse` environment provided by the package is an extension of the usual LaTeX environment. The environment takes one optional parameter, which is a length; for example `\begin{verse}[4em]`. You may have noticed that the earlier verse examples are all near the left margin, whereas verses usually look better if they are typeset about the center of the page. The length parameter, if given, should be about the length of an average line, and then the entire contents will be typeset with the mid point of the length centered horizontally on the page.

`\versewidth` The length `\versewidth` is provided as a convenience. It may be used, for example, to calculate the length of a line of text for use as the optional argument to the `verse` environment:

```
\settowidth{\versewidth}{This is the average line,}
\begin{verse}[\versewidth]
```

`altverse` (*env.*) Within the `verse` environment verses are separated by a blank line in the input. Individual verses within `verse` may, however, be enclosed in the `altverse` environment. This has the effect of indenting the 2nd, 4th, etc., lines of the verse by the length `\vgap`.

`patverse` (*env.*) Individual verses within the `verse` environment may be enclosed in the `patverse` environment. Within the environment the indentation of each line is specified by an indentation pattern, which consists of an array of digits, $d_1$ to $d_n$,

and the $n^{th}$ line is indented by $d_n$ times \vgap. However, the first line is not indented, irrespective of the value of $d_1$.

patverse* (*env.*) The patverse* environment is similar to patverse environment, except that the pattern will keep on repeating itself.

\indentpattern The indentation pattern for a patverse environment is specified via the \indentpattern{⟨*digits*⟩} command. If the pattern is shorter than the number of lines in a verse, the trailing lines will not be indented.

### 3.2.2 Other verse commands

\\ \\! * \\> Within the verse environment, the macro \\ must be used at the end of each line of a verse, except for the last line in each stanza. If the lines in a poem are to be numbered then \\! must be used at the end of the last line in each stanza (the \\ macro increments the line numbers).

The starred version, *, prohibits a page break after the line. The \\> version causes a linebreak within a verse line.

The \\ macro in its various forms can also take an optional length argument, like \\[30pt] which will insert 30pt of vertical space; in the case of \\>[30pt] an additional 30pt of horizontal space will be inserted after the linebreak (effectively \\> is shorthand for \verselinebreak).

The allowable forms of the macro are:

\\, *. \\!, \\>, \\[...], *[...], \\![...], and \\>[...].

\vin \vgap The command \vin is shorthand for \hspace{\vgap} for use at the start of an indented line of verse. The length \vgap (initially 1.5em) can be changed by \setlength or \addtolength.

\vindent When a verse line is too long to fit within the typeblock it is wrapped onto the next line with a space, given by the value of the length \vindent.

\verselinebreak Using the command \verselinebreak[⟨*length*⟩] will cause later text in the line of the verse to be typeset indented on the following line. If the optional length argument is not given the indentation is \vgap, otherwise the indentation is given by ⟨*length*⟩ plus \vgap. The broken line will count as a single line as far as the altverse and patverse environments are concerned (see also the \\> macro).

\flagverse \vleftskip Putting the command \flagverse{⟨*flag*⟩} at the start of a line of verse will typeset ⟨*flag*⟩ towards the left margin, ending a distance \vleftskip before the verse line.

### 3.2.3 Generic verse formatting

\stanzaskip The length \stanzaskip controls the spacing between stanzas. It may be changed like any other length.

\leftmargini All verse lines have a minimum indent given by the length \leftmargini which also applies to any list environment. To change the minimum indent for verses do something along the lines:

```
\newlength{\saveleftmargini}
\setlength{\saveleftmargini}{\leftmargini}
\setlength{\leftmargini}{-1em}% for example to outdent verse
% verses
\setlength{\leftmargini}{\saveleftmargini}% restore original value
```

### 3.2.4 Line numbering

\poemlines The declaration \poemlines{⟨*nth*⟩} will cause every ⟨*nth*⟩ lines of succeeding verses to be numbered. For example, \poemlines{5} will number every fifth line. The default is \poemlines{0} which prevents any numbering.

\setverselinenums The command \setverselinenums{⟨*firstlinenum*⟩}{⟨*startnumsat*⟩} can be used to set the number of the first verse line to ⟨*firstlinenum*⟩ instead of the default '1' and to specify that the first printed line number should be for line number ⟨*startnumsat*⟩. If used the command must be given within the verse environment before the first line of the verses. For example, if you were quoting portions of poems from a source where the lines were numbered, your first line might be the 112th of the original and that line was originally numbered:
\setverselinenums{112}{112}
or if it was line 115 that was first numbered:
\setverselinenums{112}{115}

Note that the numbers must be such that the following relationship holds:
firstlinenum <= startnumsat < firstlinenum + poemlines

\thepoemline \verselinenumfont Lines are numbered via \thepoemline which defaults to typesetting arabic numerals via:
\renewcommand*{\thepoemline}{\arabic{poemline}}
The particular font is defined by \verselinenumfont{⟨*font-spec*⟩}, with default:
\verselinenumfont{\rmfamily}

\vrightskip \verselinenumbersleft \verselinenumbersright By default the numbers are typeset at the distance \vrightskip into the right margin. If you want line numbers set at the left use the \verselinenumbersleft declaration. To revert to the default use \verselinenumbersright.

\label \ref The standard \label{metakey} command can be used inside the verse environment, between the end of the text of a line and the line-ending \\, to grab that line number, no matter what the setting of \poemlines. Elsewhere the standard \ref{⟨*key*⟩} command can be used to refer to the line number.

### 3.2.5 Titles

\poemtitle \poemtitle[⟨*short*⟩]{⟨*long*⟩} typesets the title of a poem and makes an entry into the ToC. There is a starred version that makes no ToC entry.

\poemtoc The kind of entry made in the ToC by the \poemtitle command is defined by \poemtoc. The initial definition is:
\newcommand{\poemtoc}{section}
for a section-like ToC entry. This can be changed to, say, chapter or subsection or ....

\poemtitlefont This macro specifies the font and positioning of the poem title. Its initial definition is:
\newcommand{\poemtitlefont}{\normalfont\bfseries\large\centering}
to give a \large bold centered title. This can of course be renewed if you want something else.

\beforepoemtitleskip \afterpoemtitleskip These two lengths are the vertical space before and after the \poemtitle title text. They are initially defined to give the same spacing as for a \section title. They can be changed by \setlength or \addtolength for different spacings.

\poemtitlemark The \poemtitle macro, but not \poemtitle*, calls the \poemtitlemark{⟨*title*⟩}

macro, which is defined to do nothing. This would probably be changed by a pagestyle definition (like `\sectionmark` or `\chaptermark`).

## 3.3 Supports

The package includes some macros for supporting the `patverse` environment which may be more generally useful. See the code section for examples on how these may be used.

`\newarray` `\newarray{`⟨*arrayname*⟩`}{`⟨*low*⟩`}{`⟨*high*⟩`}` defines the ⟨*arrayname*⟩ array, where ⟨*arrayname*⟩ is a name like `MyArray`. The lowest and highest array indices are set to ⟨*low*⟩ and ⟨*high*⟩ respectively, where both are integer numbers.

`\setarrayelement` `\setarrayelement{`⟨*arrayname*⟩`}{`⟨*index*⟩`}{`⟨*text*⟩`}` sets the ⟨*index*⟩ location in the ⟨*arrayname*⟩ array to be ⟨*text*⟩. For example: `\setarrayelement{MyArray}{23}{$2^{23}$}`.

`\getarrayelement` `\getarrayelement{`⟨*arrayname*⟩`}{`⟨*index*⟩`}{`⟨*result*⟩`}` sets the parameterless macro ⟨*result*⟩ to the contents of the ⟨*index*⟩ location in the ⟨*arrayname*⟩ array. For example: `\getarrayelement{MyArray}{23}{\result}`.

`\checkarrayindex` `\checkarrayindex{`⟨*arrayname*⟩`}{`⟨*index*⟩`}` checks if ⟨*arrayname*⟩ is an array and if ⟨*index*⟩ is a valid index for the array.

`\stringtoarray` `\stringtoarray{`⟨*arrayname*⟩`}{`⟨*string*⟩`}` puts each character from ⟨*string*⟩ sequentially into the ⟨*arrayname*⟩ array, starting at index 1. For example: `\stringtoarray{MyArray}{Chars}`.

`\arraytostring` The macro `\arraytostring{`⟨*arrayname*⟩`}{`⟨*result*⟩`}` assumes that ⟨*arrayname*⟩ is an array of characters, and defines the macro ⟨*result*⟩ to be that sequence of characters. For example: `\arraytostring{MyArray}{\MyString}`.

`\checkifinteger` `\checkifinteger{`⟨*num*⟩`}` checks if ⟨*num*⟩ is an integer (not less than zero). If it is then `\ifinteger` is set TRUE, otherwise it is set FALSE.

## 3.4 Examples

Here are some sample verses using the package facilities. First our old Limerick friend, but titled and centered:

```
\renewcommand{\poemtoc}{subsection}
\poemtitle{A Limerick}
\settowidth{\versewidth}{There was an old party of Lyme}
\begin{verse}[\versewidth]
There was an old party of Lyme \\
Who married three wives at one time. \\
\vin When asked: ‘Why the third?’ \\
\vin He replied: ‘One’s absurd, \\
And bigamy, sir, is a crime.’ \\
\end{verse}
```

which gets typeset as below. The default `\poemtoc` is redefined to `subsection` so the title is entered into the ToC as an unnumbered `\subsection`.

### A Limerick

There was an old party of Lyme
Who married three wives at one time.
When asked: 'Why the third?'
He replied: 'One's absurd,
And bigamy, sir, is a crime.'

Next is the Garden verse within the `altverse` environment. It is titled and centered.

```
\settowidth{\versewidth}{But now my love is dead}
\poemtitle{Love's lost}
\begin{verse}[\versewidth]
\begin{altverse}
\garden
\end{altverse}
\end{verse}
```

which produces:

### Love's lost

I used to love my garden
But now my love is dead
For I found a bachelor's button
In black-eyed Susan's bed.

It is left up to you how you might want to add information about the author of a poem. Here is one example of a macro for this:

```
\newcommand{\attrib}[1]{%
  \nopagebreak{\raggedleft\footnotesize #1\par}}
```

This can be used as in the next bit of doggerel.

```
\poemtitle{Fleas}
\settowidth{\versewidth}{What a funny thing is a flea}
\begin{verse}[\versewidth]
What a funny thing is a flea. \\
You can't tell a he from a she. \\
But he can. And she can. \\
Whoopee! \\
\end{verse}
\attrib{Anonymous}
```

### Fleas

What a funny thing is a flea.
You can't tell a he from a she.
But he can. And she can.
Whoopee!

Anonymous

Here is an example of line wrapping.

```
\poemtitle{In the beginning}
\settowidth{\versewidth}{And objects at rest tended to remain at rest}
\begin{verse}[\versewidth]
Then God created Newton, \\
And objects at rest tended to remain at rest, \\
And objects in motion tended to remain in motion, \\
And energy was conserved
   and momentum was conserved
   and matter was conserved \\
And God saw that it was conservative. \\
\end{verse}
\attrib{Possibly from \textit{Analog}, circa 1950}
```

### In the beginning

Then God created Newton,
And objects at rest tended to remain at rest,
And objects in motion tended to remain in motion,
And energy was conserved and momentum was conserved and
matter was conserved
And God saw that it was conservative.

Possibly from *Analog*, circa 1950

Here is one with a forced line break and a slightly different title style.

```
\renewcommand{\poemtitlefont}{\normalfont\large\itshape\centering}
\poemtitle{Mathematics}
\settowidth{\versewidth}{Than Tycho Brahe, or Erra Pater:}
\begin{verse}[\versewidth]
In mathematics he was greater \\
Than Tycho Brahe, or Erra Pater: \\
For he, by geometric scale, \\
Could take the size of pots of ale;\\ \settowidth{\versewidth}{Resolve by}
Resolve, by sines \\>[\versewidth] and tangents straight, \\
If bread or butter wanted weight; \\
And wisely tell what hour o' the day \\
The clock does strike, by Algebra. \\
\end{verse}
\attrib{Samuel Butler (1612--1680)}
```

*Mathematics*

In mathematics he was greater
Than Tycho Brahe, or Erra Pater:
For he, by geometric scale,
Could take the size of pots of ale;
Resolve, by sines
and tangents straight,
If bread or butter wanted weight;
And wisely tell what hour o' the day
The clock does strike, by Algebra.

Samuel Butler (1612–1680)

Another limerick, but this time taking advantage of the `patverse` environment and numbering every third line.

```
\settowidth{\versewidth}{There was a young lady of Ryde}
\poemtitle{The Young Lady of Ryde}
\begin{verse}[\versewidth]
\poemlines{3}
\indentpattern{00110}
\begin{patverse}
There was a young lady of Ryde \\
Who ate some apples and died. \\
The apples fermented \\
Inside the lamented \\
And made cider inside her inside. \\
\end{patverse}
\poemlines{0}
\end{verse}
```

**The Young Lady of Ryde**

There was a young lady of Ryde
Who ate some apples and died.
The apples fermented
Inside the lamented
And made cider inside her inside.

The next example is a song you may have heard of. The 'forty-niner' in line 3 refers to the gold rush of 1849.

```
\settowidth{\versewidth}{In a cavern, in a canyon,}
\poemtitle{Clementine}
\begin{verse}[\versewidth]
\poemlines{2}
\begin{altverse}
\flagverse{1.} In a cavern, in a canyon, \\
Excavating for a mine, \\
```

```
Lived a miner, forty-niner, \label{vs:49} \\
And his daughter, Clementine. \\!
\end{altverse}

\begin{altverse}
\flagverse{\textsc{chorus}} Oh my darling, Oh my darling, \\
Oh my darling Clementine. \\
Thou art lost and gone forever, \\
Oh my darling Clementine \\!
\end{altverse}
\poemlines{0}
\end{verse}
```

**Clementine**

1. In a cavern, in a canyon,
   Excavating for a mine, 2
   Lived a miner, forty-niner,
   And his daughter, Clementine. 4

CHORUS Oh my darling, Oh my darling,
   Oh my darling Clementine. 6
   Thou art lost and gone forever,
   Oh my darling Clementine 8

The last example is a much more ambitious use of `\indentpattern`. In this case it is defined as:
`\indentpattern{013555432211234689877977554565322234554445668877889 9}`
and the result is shown on the next page.

**Mouse's Tale**

Fury said to
a mouse, That
he met
in the
house,
'Let us
both go
to law:
*I* will
prosecute
*you.* —
Come, I'll
take no
denial;
We must
have a
trial:
For
really
this
morning
I've
nothing
to do.'
Said the
mouse to
the cur,
Such a
trial,
dear sir,
With no
jury or
judge,
would be
wasting
our breath.'
'I'll be
judge,
I'll be
jury.'
Said
cunning
old Fury;
'I'll try
the whole
cause
and
condemn
you
to
death.'

Lewis Carrol, *Alice's Adventures in Wonderland*, 1865

# 4 The package code

To try and avoid name clashes, all the internal commands include the string `@vs`.

## 4.1 Preliminaries

Announce the name and version of the package, which requires LaTeX 2$_\varepsilon$.

```
1 ⟨*usc⟩
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{verse}[2014/05/10 v2.4b verse typesetting]
```

For reference, here is the original definition of the `verse` environment from `classes.dtx`, based on `\let`ting `\\` equal `\@centercr`.

```
\newenvironment{verse}
  {\let\\\@centercr
   \list{}{\itemsep \z@
           \itemindent -1.5em%
           \listparindent\itemindent
           \rightmargin \leftmargin
           \advance\leftmargin 1.5em}%
  \item\relax}
  {\endlist}
```

## 4.2 Verse code

`vslineno` `poemline` `\c@fvsline` `modulo@vs` `verse@envctr` `\theHpoemlines`

We need a counter for verse lines and poem lines, and one for unique hyperref anchors (based on the verse environment). Also one for specifying the start of line numbering.

```
4 \newcounter{vslineno}
5 \newcounter{poemline}
6 \newcounter{fvsline}
7   \setcounter{fvsline}{0}
8 \newcounter{modulo@vs}
9 \newcounter{verse@envctr}\setcounter{verse@envctr}{0}
10 \providecommand{\theHpoemline}{}
11 \renewcommand*{\theHpoemline}{\arabic{verse@envctr}.\arabic{poemline}}
```

`\poemlines` `\poemlines{`⟨*nth*⟩`}` specifies that every ⟨*nth*⟩ poem line should be numbered. Default is not to number any lines.

```
12 \newcommand{\poemlines}[1]{%
13   \ifnum#1>\z@
14     \setcounter{modulo@vs}{#1}%
15   \else
16     \setcounter{modulo@vs}{0}%
17   \fi
18 }
19 \poemlines{0}
```

`\verselinenumfont` `\vlvnumfont` Set the font for line numbers.

```
20 \newcommand*{\verselinenumfont}[1]{\def\vlnumfont{#1}}
21 \verselinenumfont{\rmfamily}
```

\setverselinenums `\setverslinenums{⟨firstlinenum⟩}{⟨startnumsat⟩}` sets the number of the first verse line to be ⟨*firstlinenum*⟩ and the first line to be numbered to be ⟨*startnumsat*⟩. Note that `startnumsat < (firstlinenum + poemlines)`

```
\newcommand*{\setverselinenums}[2]{%
```

Set the poemline counter to `#1`.

```
  \setcounter{poemline}{#1}\addtocounter{poemline}{\m@ne}%
  \refstepcounter{poemline}%
  \ifnum\c@modulo@vs>\z@
```

If line numbers are to be printed, set `\c@fvsline` to a suitable value so that the first number to be printed will be line `#2`.

```
    \@tempcnta #2\relax
    \divide\@tempcnta\c@modulo@vs
    \multiply\@tempcnta\c@modulo@vs
    \c@fvsline #2\relax
    \advance\c@fvsline-\@tempcnta
  \fi
}
```

\getmodulo@vs This returns either nothing or a poem line number for printing.

```
\newcommand{\getmodulo@vs}{\bgroup
  \ifnum\c@modulo@vs<\@ne   % no line numbers
  \else
    \ifnum\c@modulo@vs<\tw@ % every line numbered
      \vlnumfont\thepoemline
    \else
      \@tempcnta\c@poemline
      \advance\@tempcnta -\c@fvsline
      \divide\@tempcnta\c@modulo@vs
      \multiply\@tempcnta\c@modulo@vs
      \advance\@tempcnta\c@fvsline
      \ifnum\@tempcnta=\c@poemline\vlnumfont\thepoemline\fi
    \fi
  \fi
\egroup}
```

\ifaltindent This should be set TRUE for indenting alternate lines.

```
\newif\ifaltindent
```

\ifpattern This should be set TRUE for indenting lines according to a pattern.

```
\newif\ifpattern
```

\ifstarpattern This should be set TRUE for indenting lines according in a `patverse*` environment.

```
\newif\ifstarpattern
```

\versewidth `\versewidth` is a convenience length for the user.

```
\newlength{\versewidth}
```

\vgap The length `\vgap` is used as the basis for spacing. `\vin` makes a horizontal space of `\vgap` and `\vindent` is the indentation of wrapped lines in a verse. `\stanzaskip` controls the space between stanzas.

\vin

\vindent

\stanzaskip

```
\newlength{\vgap}
\setlength{\vgap}{1.5em}
\newcommand{\vin}{\hspace*{\vgap}}
\newlength{\vindent}
\setlength{\vindent}{2\vgap}
\newlength{\stanzaskip}
\setlength{\stanzaskip}{0.75\baselineskip}
```

`\vleftskip` `\vrightskip` Skips to the left and right of a line of verse.

```
\newlength{\vleftskip}
\setlength{\vleftskip}{30pt}
\newlength{\vrightskip}
\setlength{\vrightskip}{10pt}
```

`\flagverse` `\flagverse{`⟨*flag*⟩`}` inserts ⟨*flag*⟩ at the left (of a line).

```
\newcommand{\flagverse}[1]{%
  \hskip-\vleftskip\llap{#1}\hskip\vleftskip
  \ignorespaces
}
```

`\verselinebreak` Break a verse line by inserting `\newline`.[1]

```
\newcommand*{\verselinebreak}[1][\z@]{%
  \newline\hspace*{#1}%
  \ignorespaces
}
```

`\incr@vsline` Increment the line counters.

```
\newcommand{\incr@vsline}{%
  \refstepcounter{poemline}%
  \stepcounter{vslineno}%
}
```

`\@vsifbang` Like the kernel `\@ifstar` except it looks for an exclamation mark!

```
\newcommand{\@vsifbang}[1]{\@ifnextchar !{\@firstoftwo{#1}}}
```

`\@vsifgt` Like the kernel `\@ifstar` except it looks for a `>` character.

```
\newcommand{\@vsifgt}[1]{\@ifnextchar >{\@firstoftwo{#1}}}
```

`\@vstypelinenumright` `\@vstypelinenumleft` `\verselinenumbersright` `\verselinenumbersleft` These control the typesetting of verse line numbers to the right and to the left of the verse. Default is to set them at the right.

```
\newcommand*{\@vstypelinenumright}{%
  \hfill\rlap{\kern\vrightskip\kern\rightmargin\getmodulo@vs}%
}
\newcommand*{\@vstypelinenumleft}{%
  \hfill\rlap{\kern-\textwidth\kern-\vrightskip\getmodulo@vs}%
}
\newcommand*{\verselinenumbersright}{\def\@vstypelinenum{\@vstypelinenumright}}
\newcommand*{\verselinenumbersleft} {\def\@vstypelinenum{\@vstypelinenumleft}}
\verselinenumbersright
```

[1] In an email to me dated 2006/01/13 Aaron Rendahl pointed out that this should include an `\ignorespaces`.

`\@vscentercr` This puts the poem line number in the margin, increments the line numbers, and then deals with the options. It is based on the kernel `\@centercr`. This has to handle various forms of the `\\` command: `\\`, `\\*`, `\\!`, and `\\>`, together with an optional length argument.

```
\newcommand{\@vscentercr}{%
  \ifhmode \unskip\else \@nolnerr\fi
  \@vstypelinenum
%%%%  \hfill\rlap{\kern\vrightskip\kern\rightmargin\getmodulo@vs}%
```

For `>` call `\verselinebreak` to process it.

```
  \@vsifgt{\verselinebreak}{%
    \incr@vsline
```

If the call is `\\*...` call `\@vsxcentercr` to handle the `*...`. If the call is `\\!`, do nothing. If the call is `\\![...]`, call `\@vsicentercr` to handle the `[...]`. Otherwise, call `\@vsxcentercr`.

```
    \par\@ifstar{\nobreak\@vsxcentercr}{%
      \@vsifbang{\@ifnextchar[ {\@vsicentercr}{}}{\@vsxcentercr}%
    }%
  }%
}
```

`\@vsxcentercr` Processes `\\*`, and either calls `\@vsicentercr` to handle a `[length]`, or `\start@vsline`.

```
\newcommand{\@vsxcentercr}{%
  \addvspace{-\parskip}%
  \@ifnextchar[ {\@vsicentercr}{\start@vsline}%
}
```

`\@vsicentercr` Processes `(\\...)[length]` and then calls `\start@vsline`.

```
\def\@vsicentercr[#1]{\vskip #1\ignorespaces \start@vsline}
```

`\start@vsline` This is called at the start of every verse line except the first.

```
\newcommand{\start@vsline}{%
  \ifaltindent\ifodd\c@vslineno\else\vin\fi\fi%
  \ifpattern\get@vsindent\fi%
  \ifstarpattern\getstar@vsindent\fi
}
```

`verse` (*env.*) The extended `verse` environment. It sets the verse line counter, then defines the particular list environment adjusting the margins to center according to the length parameter. If the length parameter is at least the `\linewidth` then the 'centering' defaults to the original `verse` layout.

```
\renewenvironment{verse}[1][\linewidth]{%
  \stepcounter{verse@envctr}%
  \setcounter{poemline}{0}\refstepcounter{poemline}%
  \setcounter{vslineno}{1}%
  \let\\=\@vscentercr
  \list{}{\itemsep \z@
          \itemindent  -\vindent%
          \listparindent\itemindent
          \parsep       \stanzaskip
          \ifdim #1 < \linewidth
```

```
            \rightmargin        \z@
            \setlength{\leftmargin}{\linewidth}%
            \addtolength{\leftmargin}{-#1}%
            \addtolength{\leftmargin}{-0.5\leftmargin}%
          \else
            \rightmargin        \leftmargin
          \fi
          \addtolength{\leftmargin}{\vindent}}%
  \item[]%
}
{\endlist}
```

`altverse` (*env.*) This sets `\altindenttrue` (afterwards false) and initialises the line counter.

```
\newenvironment{altverse}%
  {\starpatternfalse\patternfalse\altindenttrue\setcounter{vslineno}{1}}%
  {\altindentfalse}
```

## 4.3 Pattern code

The pattern code is based on the idea of converting a string of digits to an array of digits, and then being able to access the digit at a particular position in the array.

`\vs@nameedef` A shorthand for using `\protected@edef`.

```
\newcommand{\vs@nameedef}[1]{%
  \expandafter\protected@edef\csname #1\endcsname
}
```

`\ifbounderror` A flag set TRUE if an attempt is made to access an array element outside the array limits.

```
\newif\ifbounderror
```

`\ifinteger` A flag to indicate if a 'number' is an integer (TRUE) or not (FALSE).

```
\newif\ifinteger
```

`\c@chrsinstr` A counter for the number of characters.

```
\newcounter{chrsinstr}  % CHARactersINSTRing
```

`\newarray` `\newarray{`⟨*arrayname*⟩`}{`⟨*low*⟩`}{`⟨*high*⟩`}` defines an array called ⟨*arrayname*⟩ (no backslash e.g. `MyArray`), with low and high limits ⟨*low*⟩ and ⟨*high*⟩.

```
\newcommand{\newarray}[3]{%
  \vs@nameedef{#1-low}{#2}%
  \vs@nameedef{#1-high}{#3}%
  \ifnum #3<#2
    \PackageError{verse}{Limits for array #1 are in reverse order}{\@ehc}%
  \fi
}
```

`\stringtoarray` `\stringtoarray{`⟨*arrayname*⟩`}{`⟨*string*⟩`}` puts each character from ⟨*string*⟩ sequentially into the ⟨*arrayname*⟩ array, starting with ⟨*low*⟩ = 1. It checks for an empty ⟨*string*⟩ and handles that specially.

```
\newcommand{\stringtoarray}[2]{%
```

```
  \def\@vsarrayname{#1}%
  \protected@edef\the@vsstring{#2}%
  \newarray{\@vsarrayname}{1}{1}%
  \@ifmtarg{#2}{%
    \c@chrsinstr \z@
    \@namedef{\@vsarrayname-1}{}%
  }{%
    \c@chrsinstr \@ne
    \expandafter\@vsstringtoarray \the@vsstring\@vsend
  }%
}
```

`\@vsstringtoarray` Recursively adds characters to the array `\@vsarrayname`, incrementing the array's high limit.

```
\def\@vsstringtoarray #1#2\@vsend{%
  \@namedef{\@vsarrayname-\the\c@chrsinstr}{#1}%
  \vs@nameedef{\@vsarrayname-high}{\the\c@chrsinstr}%
  \@ifmtarg{#2}{%
    \def\@vsinext{}%
  }{%
    \advance\c@chrsinstr \@ne
    \def\@vsinext{%
      \@vsstringtoarray #2\@vsend%
    }%
  }%
  \@vsinext
}
```

`\setarrayelement` `\setarrayelement{`⟨*arrayname*⟩`}{`⟨*index*⟩`}{`⟨*value*⟩`}` sets the ⟨*arrayname*⟩ array's element at ⟨*index*⟩ to ⟨*value*⟩.

```
\newcommand{\setarrayelement}[3]{%
  \checkarrayindex{#1}{#2}%
  \vs@nameedef{#1-#2}{#3}%
}
```

`\getarrayelement` `\getarrayelement{`⟨*arrayname*⟩`}{`⟨*index*⟩`}{`⟨*value*⟩`}` defines the parameterless macro ⟨*value*⟩ (e.g., `\result`) to be the value at ⟨*index*⟩ in the ⟨*arrayname*⟩ array.

```
\newcommand{\getarrayelement}[3]{%
  \checkarrayindex{#1}{#2}%
  \protected@edef#3{\@nameuse{#1-#2}}%
}
```

`\checkarrayindex` `\checkarrayindex{`⟨*arrayname*⟩`}{`⟨*index*⟩`}` checks that the ⟨*index*⟩ of the ⟨*arrayname*⟩ array is valid. `\ifbounderror` is set FALSE if everything is OK, otherwise it is set TRUE.

```
\newcommand{\checkarrayindex}[2]{%
  \bounderrorfalse
  \expandafter\ifx\csname #1-low\endcsname\relax
    \ifpattern\else
      \PackageError{verse}{No array called #1}{\@ehc}%
    \fi
    \bounderrortrue
  \fi
```

```
  \ifnum #2<\@nameuse{#1-low}\relax
    \ifpattern\else
      \PackageError{verse}{Index #2 outside limits for array #1}{\@ehc}%
    \fi
    \bounderrortrue
  \fi
  \ifnum #2>\@nameuse{#1-high}\relax
    \ifpattern\else
      \PackageError{verse}{Index #2 outside limits for array #1}{\@ehc}%
    \fi
    \bounderrortrue
  \fi
}
```

`\@ifmtarg` Provides an if-then-else command for an empty macro argument (empty = zero or more spaces only). Use as:
`\@ifmtarg{arg1}{Code for arg1 empty}{Code for arg1 not empty}`
This code is copied from my ifmtarg package.

```
\begingroup
\catcode`\Q=3
\long\gdef\@ifmtarg#1{\@xifmtarg#1QQ\@secondoftwo\@firstoftwo\@nil}
\long\gdef\@xifmtarg#1#2Q#3#4#5\@nil{#4}
\long\gdef\@ifnotmtarg#1{\@xifmtarg#1QQ\@firstofone\@gobble\@nil}
\endgroup
```

`\arraytostring` `\arraytostring{`⟨*arrayname*⟩`}{`⟨*string*⟩`}` converts the characters in the ⟨*arrayname*⟩ array into the parameterless macro ⟨*string*⟩ (e.g., `\MyString`).

```
\newcommand{\arraytostring}[2]{%
  \def#2{}%
  \c@chrsinstr = \@nameuse{#1-low}%
  \@vsarraytostring{#1}{#2}%
}
```

`\@vsarraytostring` `\@vsarraytostring{`⟨*arrayname*⟩`}{`⟨*string*⟩`}` recursively adds the (character) elements from ⟨*arrayname*⟩ to ⟨*string*⟩.

```
\newcommand{\@vsarraytostring}[2]{%
  \ifnum\c@chrsinstr>\@nameuse{#1-high}\else
    \protected@edef#2{#2\@nameuse{#1-\thechrsinstr}}%
    \advance\c@chrsinstr\@ne%
    \@vsarraytostring{#1}{#2}%
  \fi%
}
```

`\checkifinteger` `\checkifinteger{`⟨*num*⟩`}` checks if ⟨*num*⟩ is an integer. If it is, then `\ifinteger` is set TRUE, otherwise it is set FALSE. (Code based on Donald Arseneau's cite package).

```
\newcommand{\checkifinteger}[1]{%
  \protected@edef\@vsa{#1}%
  \ifcat _\ifnum9<1\gobm{#1} _\else A\fi
    \integertrue
  \else
    \integerfalse
  \fi
}
```

`\gobm` `\gobm{`⟨*num*⟩`}` is defined as ⟨*num*⟩. It could be defined as:
`\newcommand{\gobm}[1]{\ifx-#1\expandafter\gobm\else#1\fi}`
which would remove a leading minus sign (hyphen) from its argument (`gobm` = gobble minus sign). (Code from a posting to CTT by Donald Arseneau on 1997/07/21).

```
\newcommand{\gobm}[1]{#1}
```

`\indentpattern` `\indentpattern{`⟨*digits*⟩`}` stores ⟨*digits*⟩ for use as a verse indentation pattern.

```
\newcommand{\indentpattern}[1]{%
  \stringtoarray{Array@vs}{#1}%
}
```

`\get@vsindent` `\get@vsindent` gets the indent pattern digit for the `\thevslineno`, then uses this to specify the line indentation as `digit*\vgap`.

```
\newcommand{\get@vsindent}{%
  \getarrayelement{Array@vs}{\number\value{vslineno}}{\@vspat}%
  \ifbounderror
    \arraytostring{Array@vs}{\@vsp@t}%
    \PackageWarning{verse}{%
      Index `\thevslineno' for pattern `\@vsp@t' is out of bounds}%
    \def\@vspat{0}%
  \else
    \checkifinteger{\@vspat}%
    \ifinteger\else
      \arraytostring{Array@vs}{\@vsp@t}%
      \PackageWarning{verse}{%
       `\@vspat' at index `\thevslineno' in pattern `\@vsp@t' is not a digit}%
      \def\@vspat{0}%
    \fi
  \fi
  \ifcase\@vspat\else\hspace*{\@vspat\vgap}\fi
}
```

`\getstar@vsindent` `\getstar@vsindent` gets the indent pattern digit for the `patverse*` environment, then uses this to specify the line indentation as `digit*\vgap`. It lets the pattern repeat by resetting the `vslineno` counter.

```
\newcommand{\getstar@vsindent}{%
  \expandafter\ifx\csname Array@vs-high\endcsname\relax
    \PackageError{verse}{A pattern has not been specified}{\@ehc}%
  \else
    \ifnum\c@vslineno>\@nameuse{Array@vs-high}%
      \setcounter{vslineno}{1}%
     \fi
     \get@vsindent
  \fi
}
```

`patverse` (*env.*) The environment for setting verse line indents according to a pattern. It starts by setting `\ifpattern` TRUE, any other flags to FALSE, and initialises the line number. It ends by setting `\ifpattern` FALSE.

```
\newenvironment{patverse}
  {\starpatternfalse\patterntrue\altindentfalse\setcounter{vslineno}{1}}
  {\patternfalse}
```

`patverse*` (*env.*) The environment for setting verse line indents according to a repeating pattern. It starts by setting `\ifstarpattern` TRUE, any other flags to FALSE, and initialises the line number. It ends by setting `\ifstarpattern` FALSE.

```
\newenvironment{patverse*}
  {\starpatterntrue\patternfalse\altindentfalse\setcounter{vslineno}{1}}
  {\starpatternfalse}
```

## 4.4 Title code

`\poemtitle` Typeset a poem title (like `\section` or other). The actual work is done by `\@vsptitle` (plain) or `\@vssptitle` (starred).

```
\newcommand{\poemtitle}{%
  \par
  \secdef\@vsptitle\@vssptitle
}
```

`\poemtoc` The kind of entry `\poemtitle` is to make in the ToC.

```
\newcommand{\poemtoc}{section}
```

`\@vsptitle` Typeset a `poemtitle`.

```
\def\@vsptitle[#1]#2{%
  \@nameuse{phantomsection}%
  \addcontentsline{toc}{\poemtoc}{#1}%
  \poemtitlemark{#1}%
  \@vstypeptitle{#2}%
  \@afterheading
}
```

`\@vssptitle` Typeset a `\poemtitle*`.

```
\def\@vssptitle#1{%
  \@vstypeptitle{#1}
  \@afterheading
}
```

`\@vstypeptitle` This *really* typesets the title.

```
\newcommand{\@vstypeptitle}[1]{%
  \vspace{\beforepoemtitleskip}%
  {\poemtitlefont #1\par}%
  \vspace{\afterpoemtitleskip}%
}
```

`\poemtitlefont` Sets the appearance to the title of a poem, and something for a header.
`\poemtitlemark`

```
\newcommand{\poemtitlefont}{\normalfont\large\bfseries\centering}
\newcommand{\poemtitlemark}[1]{}
```

`\beforepoemtitleskip` Lengths before and after a poem title, using the `\section` values.
`\afterpoemtitleskip`

```
\newlength{\beforepoemtitleskip}
\setlength{\beforepoemtitleskip}{3.5ex \@plus 1ex \@minus .2ex}
\newlength{\afterpoemtitleskip}
\setlength{\afterpoemtitleskip}{2.3ex \@plus.2ex}
```

The end of this package.

```
⟨/usc⟩
```

# References

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley Publishing Company, 1994.

[Wil01] Peter Wilson. *A Rumour of Humour: A scientist's commonplace book*. To be published.

# 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.