commalists-tools-l3

LaTeX3

Macros for manipulating numeral comma separated lists: sorting, adding, removing, etc

Version 0.1.6 – 27/08/2025

Cédric Pierquet
c pierquet – at – outlook . fr
https://github.com/cpierquet/latex-packages/tree/main//commalists-tools

---

```
\def\mytestlist{14,10,15,11,9,10}
\ctlenoflist*{\mytestlist}
\ctminoflist*{\mytestlist}
\ctmaxoflist*{14,10,15,11,9,10}
\ctsortasclist*{\mytestlist}
\ctmeanoflist*{\mytestlist}
\ctremovevalinlist*{10}{\mytestlist}
\cttestifvalinlist{15}{\mytestlist}{true}{false}
\ctgetvaluefromlist*{\mytestlist}{3}
```

We consider the list: 14,10,15,11,9,10

There’s 6 values in the list

The minimum value is 9 and the maximum is 15

Ascending sorted list is 9,10,10,11,14,15

Meaning value of the list is 11.5

If we remove the value 10, then the list is 14,15,11,9

We test if 15 is in the list: true

The third value of the list is 15

Contents

1 Loading, useful packages

In order to load `randintlist`, simply use:

```
\usepackage{commalists-tools-l3}
```

All code is written in LaTeX3, so no extra packages are needed.

2 The individual macros

2.1 Global usage

Package `commalists-tools-l3` supports (basic) manipulations on numeral comma separated lists:

- sorting;
- adding values;
- removing values;
- counting values;
- mean, sum, product;
- get value, get length;
- etc.

Starred versions only print the result, whereas non starred versions store the result into a macro.

Macros are prefixed with `\ct...` (for commalists-tools).

2.2 The macro for printing

```
%just printing, with optional separator
\ctshowlist[sep]{list}
```

```
\ctshowlist{1,   2,3, 4, 6}
```

1,2,3,4,6

```
\def\mytmplist{12, -4,   5,7 ,8, 9, 0}
\ctshowlist[\,/\,]{\mytmplist}
```

12/-4/5/7/8/9/0

2.3 The macros for calculating

```
%getting length of list, only printing
\ctlenoflist*{list}
%storing length of list into macro (\resmylen by default)
\ctlenoflist{list}
%getting min of list, only printing
\ctminoflist*{list}
%storing min of list into \macro (\resmin by default)
\ctminoflist{list}[\macro]
%getting max of list, only printing
\ctmaxoflist*{list}
%storing max of list into \macro (\resmax by default)
\ctmaxoflist{list}[\macro]
```

```
%only printing
\ctlenoflist*{14,10,15,11,9,10}\\
%only printing
\ctminoflist*{14,10,15,11,9,10} and \ctmaxoflist*{14,10,15,11,9,10}
```

6
9 and 15

```
%storing len/min/max of list
\def\mytestlist{10,14.5,20,12,8.5}
\ctlenoflist{\mytestlist}[\mylen]Length of list is \mylen\ \&
\ctminoflist{\mytestlist}[\mymin]Min of list is \mymin\ \&
\ctmaxoflist{\mytestlist}[\mymax]Max of list is \mymax
```

Length of list is 5 & Min of list is 8.5 & Max of list is 20

```
%getting mean of list, only printing
\ctmeanoflist*{list}
%storing mean of list into \macro (\resmean by default)
\ctmeanoflist{list}[\macro]
%getting sum of list, only printing
\ctsumoflist*{list}
%storing max of list into \macro (\ressum by default)
\ctsumoflist{list}[\macro]
%getting prod of list, only printing
\ctprodoflist*{list}
%storing max of list into \macro (\resprod by default)
\ctprodoflist{list}[\macro]
```

```
%only printing
\ctmeanoflist*{14,10,15,11,9,10} \\
%storing
\ctmeanoflist{14,10,15,11,9,10}[\mymean]\mymean \\
%only printing
\ctsumoflist*{14,10,15,11,9,10} and \ctprodoflist*{14,10,15,11,9,10} \\
%storing
\ctsumoflist{14,10,15,11,9,10}[\mysum]\ctprodoflist{14,10,15,11,9,10}[\myprod]
The sum is \mysum\ and the prod is \myprod
```

11.5
11.5
69 and 2079000
The sum is 69 and the prod is 2079000

2.4 Macros for manipulating

```
%sorting (asc), only printing
\ctsortasclist*{list}
%sorting (asc) and storing (overwrite)
\ctsortasclist{list}
%sorting (des), only printing
\ctsortdeslist*{list}
%sorting (des) and storing (overwrite)
\ctsortdeslist{list}
```

```
%sorting (asc), only printing
\ctsortasclist*{14,10,15,11.5,9,10}\\
%sorting (asc) and storing (overwrite)
\def\tmpsortlist{14,10,15,11.5,9,10}
\ctsortasclist{\tmpsortlist}\tmpsortlist\\
%analysing
\readlist*\tmpSORTlist{\tmpsortlist}
\showitems{\tmpSORTlist}
```

9,10,10,11.5,14,15
14,10,15,11.5,9,10
14 10 15 11.5 9 10

```
%extract value, only printing
\ctgetvaluefromlist*{list}{index}
%extract value and storing into macro (\myelt by default)
\ctgetvaluefromlist{list}{index}[\macro]
```

```
%extract value, only printing
\def\listtmp{1,2,3,6,3,1,5,7,3}%
\ctgetvaluefromlist*{\listtmp}{4}\par\smallskip
%storing
\ctgetvaluefromlist{\listtmp}{-1}[\mylastelt]The last element is \mylastelt
```

6
The last element is 3

```
%sorting (des), only printing
\ctsortdeslist*{14,10,15,11.5,9,10}\\
%sorting (asc) and storing (overwrite)
\def\tmpsortlist{14,10,15,11.5,9,10}
\ctsortdeslist{\tmpsortlist}\tmpsortlist\\
%analysing
\readlist*\tmpSORTlist{\tmpsortlist}
\showitems{\tmpSORTlist}
```

15,14,11.5,10,10,9
14,10,15,11.5,9,10
14 10 15 11.5 9 10

```
%adding, only printing
\ctaddvalinlist*{values}{list}
%adding and storing (overwrite)
\ctaddvalinlist{values}{list}
%removing, only printing
\ctremovevalinlist*{value}{list}
%removing and storing (overwrite)
\ctremovevalinlist{value}{list}
```

```
%only printing
\ctaddvalinlist*{3}{1,2,5,6}\\
%defining and adding
\def\tmpaddlist{1,2,4,5,6}
\ctaddvalinlist{3}{\tmpaddlist}\tmpaddlist\\
%analysing
\readlist*\tmpADDlist{\tmpaddlist}
\showitems{\tmpADDlist}
```

1,2,5,6,3
1,2,4,5,6,3
1 2 4 5 6 3

```
%only printing
\ctremovevalinlist*{3}{1,2,3,6,3,1,5,7,3}\\
%defining and removing
\def\tmpremlist{1,2,3,6,3,1,5,7,3}
\ctremovevalinlist{3}{\tmpremlist}\tmpremlist\\
%analysing
\readlist*\tmpREMlist{\tmpremlist}
\showitems{\tmpREMlist}
```

1,2,6,1,5,7
1,2,3,6,3,1,5,7,3
1 2 3 6 3 1 5 7 3

```
%reversing, only printing
\ctreverselist*{list}
%reversing and storing (overwrite)
\ctreverselist{list}
```

```
%only printing
\ctreverselist*{14,10,15,11,9,10}\\
%reversing and storing
%storing
\ctreverselist{14,10,15,11,9,10}[\myreverse]\myreverse\\
%analysing
\readlist*\tmpREVERSElist{\myreverse}
\showitems{\tmpREVERSElist}
```

10,9,11,15,10,14
10,9,11,15,10,14
10 9 11 15 10 14

2.5 Macros with testing

```
%testing if value is in list, with boolean result in \macro (\resisinlist by default)
\ctboolvalinlist{value}{list}[\macro]
%conditionnal test if value is in list, according to xint syntax
\cttestifvalinlist{3}{0,1,2,3}{true}{false}
```

```
%test with xint syntax
\cttestifvalinlist{-1}{0,1,2,3}{true}{false}\\
%test with xint syntax
\cttestifvalinlist{3}{0,1,2,3}{true}{false}\\
%boolean macro
\def\myteslist{0,5,10,5,6,9,7,8}
\ctboolvalinlist{5}{\myteslist}[\resisinlist]\resisinlist
```

false
true
1

```
%counting value, only printing
\ctcountvalinlist*{value}{list}
%counting value, with result in \macro (\rescount by default)
\ctcountvalinlist{value}{list}[\macro]
```

```
%only printing
\ctcountvalinlist*{8}{1,2,3,4,5,6,6,7,8,8,8,9}\\
%storing
\def\tmpcountlist{1,2,3,4,5,6,6,7,8,8,8,9}
\ctcountvalinlist{6}{\tmpcountlist}[\rescountsix]\rescountsix\\
\ctcountvalinlist{10}{\tmpcountlist}[\rescountten]\rescountten
```

3
2
0

3 History

0.1.6: Initial version,code written in LaTeX3

4 The code

```
% Author     : C. Pierquet
% licence    : Released under the LaTeX Project Public License v1.3c or later, see
     http://www.latex-project.org/lppl.txt

\NeedsTeXFormat{LaTeX2e}
\ProvidesExplPackage{commalists-tools-l3}{2025-08-27}{0.1.6}{Basic operations for numeral comma separated lists}

%------History
% 0.1.6 Initial version (prefixed macros due to legacy package)

%------Variables
\clist_new:N \l__commalists_clist
\fp_new:N \l__commalists_tmpa_fp

%-----Macros (LaTeX3...)
\NewDocumentCommand\ctshowlist{ O{,} m }
{
    \clist_set:Nx \l__commalists_clist { #2 }
    \clist_use:Nn \l__commalists_clist { #1 }
}

\NewDocumentCommand\ctsortasclist{ s m O{\mysortedlist} }
{
    \clist_set:Nx \l_foo_clist {#2}
    \clist_sort:Nn \l_foo_clist{
        \fp_compare:nNnTF { ##1 } > { ##2 }
            { \sort_return_swapped: }
            { \sort_return_same: }
    }
    \IfBooleanTF{#1}%if star := just printing // if not, storing
        {
            {\l_foo_clist}
        }
        {
            \tl_gset:Nx \l_tmpa_tl { \l_foo_clist }
            \tl_gset:Nx #3 { \tl_use:N \l_tmpa_tl }
        }
}

\NewDocumentCommand\ctsortdeslist{ s m O{\mysortedlist} }
{
    \clist_set:Nx \l_foo_clist {#2}
    \clist_sort:Nn \l_foo_clist{
        \fp_compare:nNnTF { ##1 } < { ##2 }
            { \sort_return_swapped: }
            { \sort_return_same: }
    }
    \IfBooleanTF{#1}%if star := just printing // if not, storing
        {
            {\l_foo_clist}%
        }
        {
            \tl_gset:Nx \l_tmpa_tl { \l_foo_clist }
            \tl_gset:Nx #3 { \tl_use:N \l_tmpa_tl }
        }
}

\NewDocumentCommand\ctreverselist{ s m O{\reverselist} }
{
    \clist_set:Nx \l__commalists_clist {#2}
    \IfBooleanTF{#1}%if star := just printing // if not, storing
    {
        \clist_reverse:N \l__commalists_clist
```

```
        \clist_use:Nn \l__commalists_clist { , }
    }
    {
        \clist_set_eq:NN #3 \l__commalists_clist
        \clist_greverse:N #3
    }
}

\NewDocumentCommand{\ctaddvalinlist}{ s m m }
{
    \IfBooleanTF{#1}
        {
            #3, #2
        }
        {
            \tl_gset:Nx #3 { \tl_use:N #3 , #2 }
        }
}

\NewDocumentCommand{\ctboolvalinlist}{ m m O{\resisinlist} }
{
    \clist_set:Nx \l__commalists_clist {#2}

    \tl_set:Nn #3 { 0 }

    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \tl_if_eq:nxT {##1} {#1}
        {
            \tl_gset:Nn #3 { 1 }
            \clist_map_break:
        }
    }
}

\NewDocumentCommand{\cttestifvalinlist}{ m m m m }
{
    \clist_set:Nx \l__commalists_clist {#2}

    \bool_set_false:N \l_tmpa_bool

    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \tl_if_eq:nxT {##1} {#1}
        {
            \bool_set_true:N \l_tmpa_bool
            \clist_map_break:
        }
    }
    \bool_if:NTF \l_tmpa_bool { #3 } { #4 }
}

\NewDocumentCommand{\ctremovevalinlist}{ s m m O{\mytmplist} }
{
    \clist_set:Nx \l__commalists_clist {#3}

    \clist_clear_new:N \l_tmpb_clist

    \clist_map_inline:Nn { \l__commalists_clist }
    {

        \tl_if_eq:nxF {##1} {#2}
        {
            \clist_put_right:Nn \l_tmpb_clist { ##1 }
        }
    }

    \IfBooleanTF{#1}
    {
        \clist_use:Nn \l_tmpb_clist { , }
    }
    {
```

```
        \tl_set:Nx \l_tmpa_tl { \clist_use:Nn \l_tmpb_clist { , } }
        \tl_gset:Nx #4 { \tl_use:N \l_tmpa_tl }
    }
}

\NewDocumentCommand\ctcountvalinlist{ s m m O{\rescount} }{%
    \clist_set:Nx \l__commalists_clist {#3}

    \int_zero:N \l_tmpa_int
    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \tl_if_eq:nxT {##1} {#2}
        {
            \int_incr:N \l_tmpa_int
        }
    }
    \IfBooleanTF{#1}
    {
        \int_use:N \l_tmpa_int
    }
    {
        \tl_gset:Nx #4 { \int_use:N \l_tmpa_int }
    }
}

\NewDocumentCommand\ctminoflist{ s m O{\resmin} }{%
    \IfBooleanTF{#1}%
    {
        \fp_eval:n { min(#2) }
    }
    {
        \tl_gset:Nx #3 { \fp_eval:n { min(#2) } }
    }
}

\NewDocumentCommand\ctmaxoflist{ s m O{\resmax} }{%
    \IfBooleanTF{#1}%
    {
        \fp_eval:n { max(#2) }
    }
    {
        \tl_gset:Nx #3 { \fp_eval:n { max(#2) } }
    }
}

\NewDocumentCommand\ctsumoflist{ s m O{\ressum} }{%
    \clist_set:Nx \l__commalists_clist {#2}
    \fp_set:Nn \l__commalists_tmpa_fp { 0 }
    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \fp_set:Nn \l__commalists_tmpa_fp { \fp_eval:n {\l__commalists_tmpa_fp + (##1) } }
    }
    \IfBooleanTF{#1}%
    {
        \fp_use:N \l__commalists_tmpa_fp
    }
    {
        \tl_gset:Nx #3 { \fp_use:N \l__commalists_tmpa_fp }
    }
}

\NewDocumentCommand\ctprodoflist{ s m O{\resprod} }{%
    \clist_set:Nx \l__commalists_clist {#2}
    \fp_set:Nn \l__commalists_tmpa_fp { 1 }
    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \fp_set:Nn \l__commalists_tmpa_fp { \fp_eval:n {\l__commalists_tmpa_fp * (##1) } }
    }
    \IfBooleanTF{#1}
    {
        \fp_use:N \l__commalists_tmpa_fp
    }
```

```
    {
        \tl_gset:Nx #3 { \fp_use:N \l__commalists_tmpa_fp }
    }
}

\NewDocumentCommand\ctmeanoflist{ s m O{\resmean} }{%
    \clist_set:Nx \l__commalists_clist {#2}
    \fp_set:Nn \l__commalists_tmpa_fp { 0 }
    \clist_map_inline:Nn { \l__commalists_clist }
    {
        \fp_set:Nn \l__commalists_tmpa_fp { \fp_eval:n {\l__commalists_tmpa_fp + (##1) } }
    }
    \fp_set:Nn \l__commalists_tmpa_fp { \fp_eval:n { (\l__commalists_tmpa_fp) / ( \clist_count:N {
     \l__commalists_clist } ) } }
    \IfBooleanTF{#1}%
    {
        \fp_use:N \l__commalists_tmpa_fp
    }
    {
        \tl_gset:Nx #3 { \fp_use:N \l__commalists_tmpa_fp }
    }
}

\NewDocumentCommand\ctlenoflist{ s m O{\resmylen} }{%
    \clist_set:Nx \l__commalists_clist {#2}
    \IfBooleanTF{#1}
    {
        \clist_count:N { \l__commalists_clist }
    }
    {
        \tl_gset:Nx #3 { \clist_count:N { \l__commalists_clist } }
    }
}

\NewDocumentCommand\ctgetvaluefromlist{ s m m O{\resmyelt} }{%
    \clist_set:Nx \l__commalists_clist {#2}
    \IfBooleanTF{#1}
    {
        \clist_item:Nn \l__commalists_clist { #3 }
    }
    {
        \tl_gset:Nx #4{ \clist_item:Nn \l__commalists_clist { #3 } }
    }
}

\endinput
```