Test listfun.sty

Kurt Pagani

July 29, 2022

Contents

1 Standard functions

1.1 writeList

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \writeList \lst \b \sep \e
%%% -- \lst .......... the list to display
%%% -- \b   .......... the begin character, e.g. ’[’
%%% -- \sep .......... the separator, e.g. ’, ’
%%% -- \e   .......... then end character, e.g. ’]’
%%% Writes out a list in the form specified.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

(\define \lstA (\list ’a’ ’b’ ’$\delta$’ :12 ’$\pi$’))
(\define \lstB (\list ’\LaTeX’ (\list ’$\frac{\xi}{\Omega}$’ :999) ’$\partial\Gamma$’))
%
(\texprint ’\begin{center}’)
(\writeList \lstA ’[’ ’, ’ ’]’)
(\texprint ’\end{center}’)
%
(\texprint ’\begin{center}’)
(\writeList \lstB ’[’ ’, ’ ’]’)
(\texprint ’\end{center}’)
%
(\texprint ’\begin{center}’)
(\writeList \lstA ’(’ ’ ’ ’)’)
(\texprint ’\end{center}’)
%
```

```
(\texprint ’\begin{center}’)
(\writeList \lstB ’\{’ ’ :: ’ ’\}’)
(\texprint ’\end{center}’)
```

[a, b, δ, 12, π]

[LaTeX, [$\frac{\xi}{\Omega}$, 999], $\partial\Gamma$]

(a b δ 12 π)

{LaTeX :: {$\frac{\xi}{\Omega}$ :: 999} :: $\partial\Gamma$}

1.2 equalQ

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \equalQ \x \y
%%% -- \x, \y any lisp type
%%% Tests equality recursively.
%%% Note that atomQ () -> /f, i.e () is not an atom.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\equalQ (\list) ()))
(\print (\equalQ \lstA \lstB))
(\print (\equalQ \lstA \lstA))
(\print (\equalQ \lstB \lstB))
(\print (\equalQ :123 :123))
(\print (\equalQ :0 :-0))
(\print (\equalQ ’$\alpha$’ ’$\alpha$’))
(\print (\equalQ ’A’ ’a’))
```

/t/f/t/t/t/t/t/f

1.3 atomOrNilQ

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \atomOrNilQ \x
%%% Check whether \x is an atom or the empty list (). Returns /f otherwise.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\atomQ () ))
(\print (\nilQ () ))
(\print (\atomOrNilQ () ))
(\print (\atomOrNilQ :1222))
(\print (\atomOrNilQ ’string’))
(\print (\atomOrNilQ \lstA))
```

/f/t/t/t/t/f

1.4 append

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \append \x \y
%%% Append \y to the list \x.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\define \lstAB (\append \lstA \lstB))
(\writeList \lstAB ’(’ ’, ’ ’)’)
(\texprint ’\\’) % newline
(\writeList (\append \lstA (\list ’the end of the list’)) ’[’ ’, ’ ’]’)
```

(a, b, δ, 12, π, LaTeX, ($\frac{\xi}{\Omega}$, 999), $\partial\Gamma$)
[a, b, δ, 12, π, the end of the list]

1.5 subst

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \subst \x \y \z
%%% Substitute \x for \y in the list \z.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\writeList (\subst '\copyright' '\LaTeX' \lstAB ) '[' ', ' ']')
(\texprint '\\') % newline
(\writeList (\subst 'nine' :999 \lstAB ) '[' ', ' ']')
(\texprint '\\') % newline
(\writeList (\subst (\list :1 :2 'Z') :999 \lstAB ) '[' ', ' ']')
```

[a, b, δ, 12, π, ©, [$\frac{\xi}{\Omega}$, 999], $\partial\Gamma$]
[a, b, δ, 12, π, LaTeX, [$\frac{\xi}{\Omega}$, nine], $\partial\Gamma$]
[a, b, δ, 12, π, LaTeX, [$\frac{\xi}{\Omega}$, [1, 2, Z]], $\partial\Gamma$]

1.6 memberQ

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \memberQ \x \y
%%% If \x is a member of \y then return /t else /f.
%%% Note: \x may be a sublist, and atoms are members only on first level!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\memberQ '\LaTeX' \lstAB))
(\print (\memberQ :999 \lstAB))
(\print (\memberQ '$\partial\Gamma$' \lstAB))
(\print (\memberQ (\list 'abc' :99)  (\list \lstA (\list 'abc' :99) \lstB))) % /t
(\print (\memberQ (\list '$\frac{\xi}{\Omega}$' :999) \lstAB))
```

/t/f/t/t/t

1.7 pairlis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \pairlis \x \y \a
%%% Give the list of pairs of corresponding elements of the lists \x and
%%% \y, and appends this to the list \a. The resultant list of pairs, which
%%% is like a table with two columns, is called an association list.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\define \lstPL (\pairlis (\list 'A' 'B' 'C') (\list 'U' 'V' 'W')
   (\list (\cons 'D' 'X') (\cons 'E' 'Y'))))
%(\writeList \lstPL '[' ', ' ']')
(\print \lstPL)
```

((’A’.’U’) (’B’.’V’) (’C’.’W’) (’D’.’X’) (’E’.’Y’))

1.8 assoc

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \assoc \x \a
%%% If \a is an association list, then \assoc will produce the first pair
%%% whose first term is \x. Thus it is a table searching function.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(\writeList (\assoc ’B’ \lstPL) ’[’ ’, ’ ’]’)
(\print (\assoc ’B’ \lstPL))
```

(’B’.’V’)

1.9 sublis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \sublis \a \y
%%% Here \a is assumed to be an association list of the form
%%% ((ul . v 1 ) . . . (un . v,)), where the u1’s are atomic, and \y is
%%% any S-expression. What \sublis does, is to treat the u1’s as variables
%%% when they occur in \y, and to substitute the corresponding v1’s
%%% from the pair list.
%%% Note: \sublisXXX is the helper function sub2 in the LISP 1.5 Programmer
%%% Manual (from where we have the info:).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(\define \lstSL
%  (\sublis (\list (\list ’X’ ’SHAKESPEARE’) (\list ’Y’ (\list ’THE TEMPEST’)))
%      (\list ’X’ ’WROTE’ ’Y’))
(\writeList (\list (\list ’a’ ’b’) ) ’[’ ’, ’ ’]’)
```

[[a, b]]

1.10 union

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \union \x \y
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\define \lstUN (\union (\list ’A’ ’B’ ’C’) (\list ’A’ ’X’ ’C’ ’D’)))
(\writeList \lstUN ’[’ ’, ’ ’]’)
```

[B, A, X, C, D]

1.11 intersection

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \intersection \x \y
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\define \lstIN (\intersection (\list ’A’ ’B’ ’C’) (\list ’A’ ’X’ ’C’ ’D’)))
(\writeList \lstIN ’[’ ’, ’ ’]’)
```

[A, C]

1.12 reverse

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \reverse \x
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\writeList (\reverse \lstA) ’[’ ’, ’ ’]’)
(\texprint ’\\’)
(\writeList (\reverse \lstB) ’[’ ’, ’ ’]’)
(\texprint ’\\’)
(\writeList (\reverse \lstAB) ’[’ ’, ’ ’]’)
```

[π, 12, δ, b, a]
[$\partial\Gamma$, [$\frac{\xi}{\Omega}$, 999], LaTeX]
[$\partial\Gamma$, [$\frac{\xi}{\Omega}$, 999], LaTeX, π, 12, δ, b, a]

1.13 foldr

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \foldr \f \x \y
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\texprint (\foldr (\lambda (\x \y) (\+ \x \y)) :0 (\list :1 :2 :3)))
(\texprint '\\')
(\texprint (\foldr (\lambda (\x \y) (\+ \x \y)) :1000 (\list :1 :2 :3)))
```

```
6
1006
```

1.14 foldl

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \foldl \f \x \y
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\texprint (\foldl (\lambda (\x \y) (\+ \x \y)) :1 (\list :1 :2 :3)))
(\texprint '\\')
(\texprint (\foldl (\lambda (\x \y) (\* \x \y)) :3 (\list :1 :2 :3)))
```

```
7
18
```

1.15 filter

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \filter \f \x
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\filter (\lambda (\x) (\> \x :3)) (\list :1 :2 :3 :4 :5 :6 :7)))
```

```
(:4 :5 :6 :7)
```

1.16 allQ

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \allQ \f \x
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\allQ (\lambda (\x) (\> \x :3)) (\list :1 :2 :3 :4 :5 :6 :7)))
(\texprint '\\')
(\print (\allQ (\lambda (\x) (\< \x :10)) (\list :1 :2 :3 :4 :5 :6 :7)))
```

```
/f
/t
```

1.17 anyQ

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% \anyQ \f \x
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
(\print (\anyQ (\lambda (\x) (\< \x :3)) (\list :1 :2 :3 :4 :5 :6 :7)))
(\texprint '\\')
(\print (\anyQ (\lambda (\x) (\> \x :10)) (\list :1 :2 :3 :4 :5 :6 :7)))
```

```
/t
/f
```

/t/t/t('A'.'B')