

The `ran_toks` Package

D. P. Story
Email: `dpstory@uakron.edu`

processed June 6, 2021

Contents

1	Alternate package name: <code>ran-toks</code>	3
2	Commands for controlling the process	3
3	Utility commands	4
4	The main commands	7
4.1	Additional user access commands	10
5	A DB application	12
6	Index	18
7	Change History	20

1 `\package`

Description. This short package randomizes a list of tokens. The command, `\ranToks`, takes one argument, which is a list of tokens:

```
\ranToks{name}{%  
  {tok1}{tok2}...{tokn}  
}
```

The command defines a series of n internal commands, one for each of the tokens. The definitions are essentially randomized. The randomized tokens are accessed through the command `\useRanTok`. For example,

```
\useRanTok{1}, \useRanTok{2}, ..., \useRanTok{n}
```

gives a random listing of the n tokens. These can be arranged on the page as desired.

There is a second construct, designed for more elaborate randomization.

```

\brTVToks{<name>}
\begin{rtVW}
  <some content>
\end{rtVW}
...
...
\begin{rtVW}
  <some content>
\end{rtVW}
\erTVToks

```

The contents of each of the `rtVW` environments are written to the computers hard drive, then input back in random order, using `\useRanTok`, eg,

```
\useRanTok{1}, \useRanTok{2}, ..., \useRanTok{n}
```

Other details are left to the readers' imagination.

Requirements. As of this writing, we require only the `verbatim` package and `random.tex`, the package was written by Donald Arseneau.

```
2 \RequirePackage{verbatim}
```

Input `random.tex`. Input `random.tex` if not already input.

```
3 \@ifundefined{nextrandom}{\input{random.tex}}{}
```

We redefine `\nextrandom` from `random.tex` to save the initializing seed.

```

4 \def\nextrandom{\begingroup
5   \ifnum\randomi<\@ne % then initialize with time
6     \global\randomi\time
7     \global\multiply\randomi388 \global\advance\randomi\year
8     \global\multiply\randomi31 \global\advance\randomi\day
9     \global\multiply\randomi97 \global\advance\randomi\month
10    \message{Randomizer initialized to \the\randomi.}%
11    \nextrandom \nextrandom \nextrandom

```

Save the initial seed value to `\rtInitSeedValue`.

```

12   \xdef\InitSeedValue{\the\randomi}%
13 \fi
14 \count@ii\randomi
15 \divide\count@ii 127773 % modulus = multiplier * 127773 + 2836
16 \count@\count@ii
17 \multiply\count@ii 127773
18 \global\advance\randomi-\count@ii % random mod 127773
19 \global\multiply\randomi 16807
20 \multiply\count@ 2836
21 \global\advance\randomi-\count@
22 \ifnum\randomi<\z@ \global\advance\randomi 2147483647\relax\fi
23 \endgroup
24 }

```

The code for this package was taken from the `dps` package, and modified suitably. We use several token registers and count registers. This can probably be optimized.

```

25 \newtoks\rt@listIn \rt@listIn={}
26 \newtoks\rt@newListIn \rt@newListIn={}
27 \newtoks\rt@listOut \rt@listOut={}
28 \newcount\rt@nMax
29 \newcount\rt@nCnt
30 \newcount\rt@getRanNum
31 \newif\ifrtdebug \rtdebugfalse
32 \newif\ifwerandomize \werandomizetrue
33 \newif\ifsaveseed\saveseedtrue
34 \newif\ifrt@InputUsedIDs\rt@InputUsedIDsfalse
35 \newwrite\rt@Verb@write

```

Convenience commands.

```

36 \def\rtcsarg#1#2{\expandafter#1\csname#2\endcsname}
37 \def\rt@nameedef#1{\expandafter\edef\csname #1\endcsname}

```

`usedbapp` The code to support a DB application has grown, so much so, it deserves a option so as to include the code only if needed.

```

38 \DeclareOption{usedbapp}{\let\rtPkgInpt\rt@PkgInpt}
39 \def\rt@PkgInpt{\InputIfFileExists{rt-dbapp.def}
40  {\PackageInfo{ran_toks}{Inputting rt-dbapp.def}}
41  {\PackageInfo{ran_toks}{Cannot find rt-dbapp.def}}
42 }
43 \let\rtPkgInpt\relax
44 \AtEndOfPackage{\rtPkgInpt}
45 \ProcessOptions\relax
46 </package>
47 <*altpkgname>

```

1 Alternate package name: ran-toks

CTAN lists this package (`ran_toks`) as `ran-toks`, so we'll create a dummy package by that name.

```

48 \NeedsTeXFormat{LaTeX2e}
49 \ProvidesPackage{ran-toks}
50 [2019/12/28 v1.0 ran-toks Alt-name (dps)]
51 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{ran_toks}}
52 \ProcessOptions
53 \RequirePackage{ran_toks}[2019/12/28]
54 </altpkgname>
55 <*package>

```

2 Commands for controlling the process

`\ranToksOn` These two turn on and turn off randomization.
`\ranToksOff`

```

56 \def\ranToksOn{\werandomizetrue}
57 \def\ranToksOff{\werandomizefalse}
\useThisSeed initializes the random number generator. Use this to reproduce the same sequence
of pseudo-random numbers from an earlier run. We also set \saveSeedfalse so
we do not write the initial seed to the disk.
58 \def\useThisSeed#1{\saveSeedfalse\randomi=#1}
59 \@onlypreamble\useThisSeed
\useLastAsSeed initializes the random number generator using the last random seed. If the file
\jobname_rt.sav does not exist, the generator will be initialized using time and
date data.
60 \def\useLastAsSeed{\rt@useLastAsSeed}
61 \@onlypreamble\useLastAsSeed
62 \def\rt@useLastAsSeed{%
63   \IfFileExists{\jobname_rt.sav}{%
64     \PackageInfo{ran_toks}{Inputting \jobname_rt.sav}%
65     \@ifundefined{readsavfile}{\newread\readsavfile}{}%
66     \openin\readsavfile=\jobname_rt.sav
67     \read\readsavfile to \InitSeedValue
68     \read\readsavfile to \lastRandomNum
69     \closein\readsavfile
70     \randomi=\lastRandomNum
}
When \useLastAsSeed, the last becomes the first.
71   \xdef\InitSeedValue{\the\randomi}
72 }{%
73   \PackageInfo{ran_toks}{\jobname_rt.sav cannot
74     be found, \MessageBreak
75     using the random initializer}%
76 }%
77 }
78 \@ifundefined{aeb@randomizeChoices}{%
79   \let\inputRandomSeed\useLastAsSeed
80   \let\useRandomSeed\useThisSeed}{%

```

3 Utility commands

A standard `\verbatim` write used in `exerquiz` and other package in the AeB family.

```

81 \def\verbatimwrite{\@bsphack
82   \let\do\@makeother\dospecials
83   \catcode'\^^M\active \catcode'\^^I=12
84   \def\verbatim@processline{%
85     \immediate\write\verbatim@out
86     {\the\verbatim@line}}%
87   \verbatim@start}
88 \def\endverbatimwrite{\@esphack}
89 \def\rt@IWVO{\immediate\write\verbatim@out}

```

We write only if `\ifsaveSeed` is true.

```

90 \def\InitSeedValue{\the\randomi}
91 \def\rt@writeSeedData{\ifsaveSeed
92   \@ifundefined{saveSeedInfo}{\newwrite\saveSeedInfo}{}
93   \immediate\openout \saveSeedInfo \jobname_rt.sav
94   \let\verbatim@out\saveSeedInfo
95   \def\rt@msgi{initializing seed value}%
96   \def\rt@msgii{last random number used}%
97   \uccode'c='\% \uppercase{%
98   \rt@IWV0{\InitSeedValue\space c \rt@msgi}%
99   \rt@IWV0{\the\randomi\space c \rt@msgii}}\immediate
100  \closeout\saveSeedInfo\fi}

```

Save the initial seed value to hard drive.

```
101 \AtEndDocument{\rt@writeSeedData}%
```

`\rt@populateList{<n>}` is a utility command, its argument `<n>` is a positive integer, and it generates a list of the form `\{1\}\{2\} . . . \{n\}` and is held in the token register `\rt@listIn`. This listing is later randomly permuted by `\rt@RandomizeList`.

```

102 \def\rt@populateList#1{\rt@listIn={}\rt@nCnt\z@
103   \@whilenum\rt@nCnt<#1\do{\advance\rt@nCnt\@ne
104     \edef\rt@listInHold{\the\rt@listIn\noexpand\{\the\rt@nCnt}}%
105     \rt@listIn=\expandafter{\rt@listInHold}}

```

`\rt@RandomizeList{<n>}` is the command that gets the process of randomizing the input list going. The argument is the number `<n>` of tokens. If `\werandomize` is false, it just returns the input list; otherwise, it calls `\rt@randomizeList` to actually do the work.

```

106 \def\rt@RandomizeList#1{\global
107   \rt@listIn={}\global\rt@newListIn={}\global\rt@listOut={}%
108   \rt@nMax=#1\relax\rt@populateList{\the\rt@nMax}%
109   \ifwerandomize
110     \expandafter\rt@randomizeList\else
111     \global\rt@listOut=\expandafter{\the\rt@listIn}\fi

```

Save the list out as `\rt@BaseName-List` for later retrieval. This is the randomized list of integers for this base name.

```
112 \global\rt@nameedef{\rt@BaseName-List}{\the\rt@listOut}}
```

`\rt@randomizeList` randomizes the list of consecutive integers, and leaves the results,

$$\{k_1\}\{k_2\} . . . \{k_n\}$$

in the token register `\rt@listOut`. `\rt@randomizeList` is a loop, looping between itself and `\rt@loopTest`.

```

113 \def\rt@randomizeList{\let\=\rt@processi
114   \setranum{\rt@getRanNum}{1}{\the\rt@nMax}%
115   \ifrtdebug\typeout{\string\rt@getRanNum=\the\rt@getRanNum}\fi
116   \rt@nCnt\z@
117   \ifrtdebug\typeout{LISTING: \the\rt@listIn}\fi
118   \the\rt@listIn
119   \rt@loopTest

```

```

120 }
121 \def\rt@loopTest{\advance\rt@nMax\m@ne
122   \ifnum\rt@nMax>\z@
123     \def\rt@next{%
124       \rt@listIn=\expandafter{\the\rt@newListIn}%
125       \rt@newListIn={}\rt@randomizeList}%
126   \else
127     \let\rt@next\relax
128     \global\rt@listOut=\expandafter{\the\rt@listOut}%
129     \ifrtdebug
130       \typeout{Final Result: \string\rt@listOut=\the\rt@listOut}\fi
131   \fi\rt@next
132 }

```

In `\rt@randomizeList`, we `\let\=\rt@processi` before dumping the contents of `\rt@listIn`. We then go into a loop `\rt@loopTest`. `\rt@getRanNum` is the random integer between 1 and `\rt@nMax`.

```

133 \def\rt@processi#1{\advance\rt@nCnt\@ne
134   \ifnum\rt@nCnt=\rt@getRanNum
135     \edef\rt@listOutHold{\the\rt@listOut}%
136     \global\rt@listOut=\expandafter{\rt@listOutHold\{\#1}}%
137     \ifrtdebug\typeout{Found it: \string\{\#1}}%
138     \typeout{New \string\rt@listOut: \the\rt@listOut}\fi
139   \else
140     \edef\rt@listInHold{\the\rt@newListIn}%
141     \rt@newListIn=\expandafter{\rt@listInHold\{\#1}}%
142     \ifrtdebug\typeout{\string\rt@newListIn: \the\rt@newListIn}\fi
143   \fi
144 }

```

We perform modular arithmetic when the index of `\useRanTok` is too large. `\rt@modarith` performs modular arithmetic on its arguments (`#1 mod #2`) and returns the result in the macro `\rt@mod`.

```

145 \def\rt@modarith#1#2{\count\z@=#1\relax\count\tw@=#1\relax
146   \advance\count\z@\m@ne\divide\count\z@ #2\relax
147   \multiply\count\z@ #2\relax
148   \advance\count\tw@-\count\z@
149   \edef\rt@mod{\the\count\tw@}}

```

This macro uses `\dimen0` and `\dimen2`, so it should be called within a group.

```

\rt@badIndex   Warning messages, these are \rt@badIndex and \rt@badTokName.
\rt@badTokName 150 \def\rt@badIndex#1#2{\PackageWarningNoLine{ran_toks}
151   {The argument of \string\useRanTok{#1} on line
152     \the\inputlineno\space is\MessageBreak
153     greater than \string\nToksFor{#2} (\nToksFor{#2}),
154     instead will use\MessageBreak
155     \string\useRanTok{\rt@mod}, obtained from modular
156     arithmetic.\MessageBreak
157     You might want to fix this}%
158 }

```

```

159 \def\rt@badTokName#1{%
160   \PackageWarningNoLine{ran_toks}
161     {The token list ‘#1’ on line \the\inputlineno\space
162     is undefined,\MessageBreak
163     possibly simply misspelled; check spelling.\MessageBreak
164     If undefined, use \string\ranToks\space or \string\bRTVToks/%
165     \string\erTVToks\space\MessageBreak
166     to define a list with the name ‘#1’}%
167 }
168 \def\rt@warnTokName#1{%
169   \PackageWarningNoLine{ran_toks}
170     {The token list ‘#1’ on line \the\inputlineno\space
171     is already defined,\MessageBreak
172     will overwrite this list}%
173 }

```

4 The main commands

`\ranToks`{*token-list*} takes one argument, *{token-list}*, a list of tokens. It randomizes them. The randomized listing can be accessed using `\useRanTok`.

```

174 \def\ranToks#1{\begingroup
175   \useRTName{#1}%
176   \r@nToks
177 }
178 \long\def\r@nToks#1{\rt@nMax\z@\r@endToks#1\rt@NIL}
179 \def\rt@NIL{@nil}

```

`\useRTName`{*name*} sets the base name (use prior to the use of `\useRanTok`).

```

180 \newcommand{\useRTName}[1]{\gdef\rt@BaseName{#1}}%
181 \let\rt@BaseName\@empty

```

`\bRTVToks`{*name*} `\bRTVToks` and `\erTVToks` enclose a series of `rtVW` environments. The single argument is the name of this set of verbatim write “tokens”.

```

182 \newcommand{\bRTVToks}[1]{\rt@nCnT\z@\useRTName{#1}}

```

`\erTVToks` At the end of the `rtVW` environments, initiated by `\bRTVToks`, the `\erTVToks` command saves the number of tokens counted, and randomizes the access to the contents of the `rtVW` environments, this done by `\r@nVToks`.

```

183 \newcommand{\erTVToks}{%
184   \global\rt@nameedef{\rt@BaseName Cnt}{\the\rt@nCnT}\expandafter
185   \r@nVToks\expandafter{\rt@BaseName}}

```

`rtVW` `\rtVW` is a verbatim write environment. It saves its contents to the file `\jobname_\rt@BaseName-\the\rt@nCnT.cut`. The file is later input back into the source file in a random way.

```

186 \def\reVerbEnd{\ifhmode\unskip\fi}

```

Insert the hook `\rtVWHook` prior to writing the verbatim content. The default is `\relax`.

```

187 \def\rtVWHook#1{\def\@rgi{#1}\ifx\@rgi\@empty
188 \let\RTVWHook\relax\else\def\RTVWHook{#1}\fi}
189 \rtVWHook{}
190 \newwrite\wrtprobid
191 \newif\ifviewIDs\viewIDsfalse
192 \newif\ifxDBUnique\xDBUniquefalse
193 \def\wrtProbIds#1{\immediate\write\wrtprobid{\string
194 \rtcsarg\string\gdef{#1}{used}}}
195 \def\rtVWId#1{\ifviewIDs\noindent#1\fi
196 \ifxDBUnique\ifrt@InputUsedIDs\wrtProbIds{#1}\fi\fi
197 }

198 \newenvironment{rtVW}{\global\advance\rt@nCt\@ne
199 \immediate\openout\rt@Verb@write
200 \jobname_\rt@BaseName-\the\rt@nCt.cut
201 \let\verbatim@out\rt@Verb@write
202 \rt@IWVO{\string\RTVWHook}%
203 \rt@IWVO{\string
204 \rtVWId{\rt@BaseName-\the\rt@nCt}\string\relax}%
205 \verbatimwrite
206 }{%
207 \endverbatimwrite
208 \immediate\write\rt@Verb@write{\string\reVerbEnd}%
209 \immediate\closeout\rt@Verb@write
210 }

```

(2021/05/29) In support of nested `\bRTVToks`/`\eRTVToks` constructs, we `\let` `\rtVWi` and `\rtVWii` to `\rtVW`, and `\endrtVWi` and `\endrtVWii` to `\endrtVW`.

```

211 \let\rtVWi\rtVW % dps5-29
212 \let\endrtVWi\endrtVW
213 \let\rtVWii\rtVW % dps5-29
214 \let\endrtVWii\endrtVW

```

`\r@nVToks` randomizes the contents of the `rtVW` environment.

```

215 \def\r@nVToks#1{\begingroup
216 \gdef\rt@BaseName{#1}%
217 \expandafter\rt@nMax\@nameuse{#1Ct}%
218 \rt@listIn={}\rt@nCt=0\relax\let\rt@listInHold\@empty
219 \@whilenum\rt@nCt<\rt@nMax\do{\advance\rt@nCt\@ne
220 \edef\rt@listInHold{%
221 \the\rt@listIn{\noexpand\rt@inputVerb{#1-\the\rt@nCt}}}% J14
222 \rt@listIn=\expandafter{\rt@listInHold}\ifrtdebug
223 \typeout{\string\r@nVToks: \the\rt@listIn}\fi
224 \expandafter\r@nToks\expandafter{\the\rt@listIn}}

```

`\rt@inputVerb{<db-name>-<num>}` This is the command that inputs a DB Test problem, it inputs it from the file named `\jobname-<db-name>-<num>`. As we input, we make a record of the problem we input by expanding `\rt@recordAsUsed{<db-name>-<num>}`,

which itself expands to `\rtcsarg\gdef{<db-name>-<num>}{used}`. This is necessary when we are choosing more than one item from a given DB Test file; it must be recorded immediately so that later it cannot be used again, if possible.

```
225 \def\rt@inputVerb#1{\rt@recordAsUsed{#1}\input{\jobname_#1.cut}}
226 %\def\rt@recordAsUsed#1{\ifxDBUnique\rtcsarg\gdef{#1}{used}\fi}
227 \def\rt@recordAsUsed#1{\rtcsarg\gdef{#1}{used}}
```

`\rt@endToks` is main looping command for `\ranToks` and `\eRTVToks` (through `\r@nVToks`). If the ending token `\rt@NIL` is detected, we break off and go to `\rt@endToks`.

```
228 \def\rt@PAR{\par}
229 \long\def\rt@endToks#1{\def\rt@rgi{#1}%
  If the current argument is \par, we skip it
230 \ifx\rt@rgi\rt@PAR\def\rt@next{\rt@endToks}\else
231 \advance\rt@nMax\@ne
232 \global\@namedef{rtTok\the\rt@nMax\rt@BaseName}{#1}%
233 \def\rt@next{\@ifnextchar\rt@NIL
234 {\rt@endToks\@gobble}{\rt@endToks}}\fi\rt@next}
```

`\rt@performRanDefns{<n>}` The `\rt@performRanDefns` performs code that is repeated in several other macros: `\rt@endToks`, `\reorderRanToks`, and `\copyRanToks`. It randomizes the list `\rt@RandomizeList`, then assignments the randomized list to the definitions.

```
235 \def\rt@performRanDefns#1{%
  Now we randomize the order of the integers 1, 2, ... #1.
236 \rt@RandomizeList{#1}\rt@nCnt\z@
  Now we randomize the definitions. We \let\=\rt@ssign, then let loose the
  tokens!
237 \let\=\rt@ssign\the\rt@listOut}
```

`\rt@endToks` The final destination for `\rt@endToks`.

```
238 \def\rt@endToks{\global
  Save the number of tokens counted
239 \rt@nameedef{nMax4\rt@BaseName}{\the\rt@nMax}%
240 \rt@performRanDefns{\the\rt@nMax}\endgroup}
```

`\reorderRanToks{<name>}` The `\reorderRanToks` command reorders (or re-indexes) the family with name `<name>` (`#1`).

```
241 \def\reorderRanToks#1{\begingroup\useRTName{#1}\expandafter
242 \ifx\csname nMax4#1\endcsname\relax
  Document author has not run \ranToks yet for this basename (#1)
243 \rt@badTokName{#1}\else
  Good to go. We reorder this list.
244 \rt@performRanDefns{\@nameuse{nMax4#1}}\fi
245 \endgroup}
```

`\copyRanToks{<name1>}{<name2>}` Use this command to copy *<name1>* to *<name2>*. This gives a randomization of the same list, without affecting the original order of *<name1>*.

```
246 \newcommand\copyRanToks[2]{\begingroup
247 \expandafter
248 \ifx\csname nMax4#1\endcsname\relax
```

Source list is not defined

```
249 \rt@badTokName{#1}%
250 \else
```

Source list is defined

```
251 \expandafter
252 \ifx\csname nMax4#2\endcsname\relax
```

Destination list is not defined, which is good in this instance. This is the case we copy the list.

```
253 \useRTName{#2}\global
254 \rt@nameedef{nMax4#2}{\@nameuse{nMax4#1}}%
255 \rt@nCt=\csname nMax4#2\endcsname\relax
256 \@whilenum\rt@nCt>\z@\do{\global
257 \rt@nameedef{rtTok\the\rt@nCt#2}%
258 {\noexpand\@nameuse{rtTok\the\rt@nCt#1}}%
259 \advance\rt@nCt\m@ne
260 }\rt@performRanDefns{\@nameuse{nMax4#2}}%
261 \else
```

Destination list is defined already, warn the user.

```
262 \rt@warnTokName{#2}%
263 \fi
264 \fi
265 \endgroup
266 }
```

`\rt@ssign{<name>}` makes the assignments that are expanded by `\useRanTok`. We `\let` the assignment `\let\=\rt@ssign` in `\rt@endToks`, just before we dump out the contents of `\the\rt@listOut`.

```
267 \def\rt@ssign#1{\advance\rt@nCt\@ne\global
268 \rt@nameedef{rtRanTok\the\rt@nCt\rt@BaseName}{\noexpand
269 \@nameuse{rtTok#1\rt@BaseName}}}
```

4.1 Additional user access commands

`\nToksFor{<name>}` expands the the number of tokens whose name is *<name>* (*#1*).

```
270 \newcommand{nToksFor}[1]{\expandafter
271 \ifx\csname nMax4#1\endcsname\relax
272 \textbf{??}\rt@badTokName{#1}\else
273 \@nameuse{nMax4#1}\fi
274 }
```

`\rtTokByNum[<name>]{<num>}` is an internal macro, but it can be used publicly. The argument of it is an integer, eg, `\rtTokByNum{3}` is the third token, as listed in the order given in the argument of `\ranToks`.

```

275 \newcommand{\rtTokByNum}[2][\rt@BaseName]{\expandafter
276   \ifx\csname nMax4#1\endcsname\relax
277     \textbf{??}\rt@badTokName{#1}\else
278     \@nameuse{\rtTok#2#1}\expandafter\ignorespaces
279   \fi
280 }

```

`\useRanTok[<name>]{<num>}` After `\ranToks` has been executed, the user has access to the randomized tokens through `\useRanTok`. The argument *<num>* is an integer 1 through max.

We provide two commands to control the feature of try to select unique choices across multiple renditions of the same source file. `\uniqueXDBChoicesOn`, turns on this feature; the default is `\uniqueXDBChoicesOff` make no changes to how `\useRanTok` operates. One other command we define here is `\makeInfoAWarning`; this command applies only when `\uniqueXDBChoicesOn` is expanded. In the macro `\xdb@unique` which is expanded when `\uniqueXDBChoicesOn` is expanded first, there is one line that reports information to the log as `\PackageInfo`. By expanding `\makeInfoAWarning` we change `\PackageInfo` to `\PackageWarning`, which gives it greater visibility in the log and the log report.

```

\uniqueXDBChoicesOn
\uniqueXDBChoicesOff
\makeInfoAWarning

```

```

281 \newcommand{\uniqueXDBChoicesOn}{\xdbUniquefalse
282   \PackageWarning{ran_toks}
283   {The \string\uniqueXDBChoicesOn\space requires the\MessageBreak
284   \texttt{usedbapp} option}}
285 \newcommand{\uniqueXDBChoicesOff}{\let\xdbunique\relax\xdbUniquefalse}
286 \let\xdbunique\relax
287 \newcommand{\makeInfoAWarning}{\def\pkgNotifType{\PackageWarning}}
288 \def\pkgNotifType{\PackageInfo}
289 \newif\ifrt@recording \rt@recordingtrue % dps

```

Now for the definition of `\useRanTok`.

```

290 \newcommand{\useRanTok}[2][\rt@BaseName]{\bgroup\expandafter
291   \ifx\csname nMax4#1\endcsname\relax
292     \rt@badTokName{#1}\global\let\rt@next\relax
293   \else
294     \ifnum#2>\@nameuse{nMax4#1}%
295       \rt@modarith{#2}{\@nameuse{nMax4#1}}%
296       \rt@badIndex{#2}{#1}\edef\Indx{\rt@mod}%
297     \else
298       \edef\Indx{#2}%
299     \fi
300     \xdef\rt@orig@Indx{\Indx}%

```

If `\xdbunique` is `\relax`, `\useRanTok` executes as it did in the past (no change in behavior); otherwise, we expand `\xdb@unique` which attempts to avoid duplicate choices based on the DBs input by `\useProbDBs`.

```

301   \ifx\xdbunique\relax

```

```

302     \ifrt@recording\rt@recordAsUsed{#1-\Indx}\fi
303     \xdef\rt@next{\noexpand\@nameuse{rtRanTok\Indx#1}}%
304   \else
305     \xdb@unique{#1}%
306   \fi
307 \fi
308 \egroup
309 \rt@next
310 }

```

`\displayListRandomly` [*⟨prior⟩*] [*⟨post⟩*] {*⟨name⟩*} lists all items in the list as passed by the required argument. For expanding in a list environment, use `\item` as the optional argument. Designed for listing all question in an eqexam document in random order.

```

311 \newcommand{\displayListRandomly}[1] [] {\begingroup
312   \def\rt@prior{#1}\displ@yListRandomly
313 }
314 \newcommand{\displ@yListRandomly}[2] [] {\@tempcntb\z@ % dps5-29
315   \expandafter\ifx\csname nMax4#2\endcsname\relax
316     \rt@rgi\space\textbf{??}\rt@badTokName{#2}#1%
317   \else
318     \rt@recordingfalse

```

```

\i   Within the optional arguments, we define \i, \first, \last, and \lessone to
\first do some logic on the arguments. These four macro are defined locally and not
\last available outside the command \displayListRandomly.
\lessone
319   \def\rt@post{#1}\useRTName{#2}\let\i\@tempcntb
320   \def\first{1}\edef\last{\@nameuse{nMax4#2}}\@tempcnta\last
321   \advance\@tempcnta\m@ne
322   \edef\lessone{\the\@tempcnta}\@whilenum\@tempcntb<\last
323     \advance\@tempcntb\@ne

```

There is one example of this command getting confused and printing the wrong number of items. Here, we pass the optional argument of `\useRanTok` and that cleared up the problem.

```

324     \do{\rt@prior\useRanTok[#2]{\the\@tempcntb}\rt@post
325   }\fi
326 \endgroup
327 }
328 </package>
329 <*dbapp>

```

5 A DB application

This (optional) section supports an application of `ran_toks` to the `eqexam` package; though, conceptually, the commands of this section may be applied in other settings. In this application, the document author has a series of DB test files (TEX files), each file contains `\bRTVToks`/`\eRTVToks` constructs, which contain a series of `rtVW` environments of verbatim content. In this application, the verbatim content are `problem/problem*` environments of `eqexam`.

The following verbatim listing is taken from the preamble of `mc-db.tex`, which illustrates the layout of how to apply the commands of this section.

```

\examNum{1}
\numVersions{4}
\forVersion{a}
% initial seeds for each of the four versions of this document
\VA{\useThisSeed{54356}}
\VB{\useThisSeed{577867}}
\VC{\useThisSeed{6746788}}
\VD{\useThisSeed{856785}}

\uniqueXDBChoicesOn % Try to avoid duplicate questions in multi-version doc
\InputUsedIDs      % Input history of previous versions to current version
\viewIDstrue      % To view the IDs of problems used
...
\useTheseDBs{db1,db2,db3,db4}

```

If `\ifxDBUnique` is true and if `eqexam` is loaded, we open `\wrtprobids` which is used to write the problem IDs of the problems already chosen in earlier version of this source file. The name of this file is `\jobname-ver\selVersion.cut`; eg, `mc-db-verA.cut`, `mc-db-verB.cut`, etc.

```

330 \def\rt@OpenProbIds{\ifpackageloaded{eqexam}
331  {\immediate\openout\wrtprobids\jobname-ver\selVersion.cut}{}}

```

We open the file `\jobname-ver\selVersion.cut` when `\InputUsedIDs` is expanded in the preamble.

```

332 %\def\rt@ABD{\ifxDBUnique\expandafter\rt@OpenProbIds\fi}
333 \def\rt@ABD{\@ifundefined{eq@nVersions}}{}
334  {\ifnum\eq@nVersions>\@ne\expandafter\rt@OpenProbIds\fi}}

```

We begin with some utility commands to help parse the argument of `\useProbDBs`.

```

335 \def\rt@gettonil#1\@nil{\def\to@nilarg{#1}}
336 \def\rt@ifspc{\ifx\@let@token\@sptoken
337  \let\rt@next\rt@xifspc\else
338  \let\rt@next\rt@gettonil\fi\rt@next
339 }
340 \begingroup
341 \def\:\{\rt@xifspc}\expandafter
342 \gdef\:\{\futurelet\@let@token\rt@ifspc}
343 \endgroup
344 \def\rt@strpspc{\futurelet\@let@token\rt@ifspc}

```

`\useTheseDBs{\list}` Inputs any files included in the comma-delimited list. The base names need only be listed, as the extension is assumed to be `.tex`. The command `\useProbDBs` can only be used in the preamble. Refer to the demo file `mc-db.tex` for an illustration of its intended use.

```

345 \def\ProbDBWarningMsg#1{\filename@parse{#1}
346  \PackageWarning{ran_toks}
347  {The file \filename@area\filename@base.\ifx\filename@ext\relax

```

```

348     tex\else\filename@ext\fi\space cannot be found}}
349 \def\useTheseDBs#1{\def\rt@dblist{#1}\ifx\rt@dblist\@empty\else
350   \let\rt@DB@List\@empty
351   \edef\temp@expand{\noexpand\@for\noexpand\@tmp:=\rt@dblist}%
352   \temp@expand\do{\ifx\@tmp\@empty\else
353     \expandafter\rt@strpspcs\@tmp\@nil\edef\@tmp{\to@nilarg}%
354     \edef\rt@nextDB{\noexpand
355       \InputIfFileExists{\@tmp}{}\noexpand
356       \ProbDBWarningMsg{\@tmp}}}%
357     \toks\tw=\expandafter{\rt@DB@List}%
358     \toks@=\expandafter{\rt@nextDB}%
359     \edef\rt@DB@List{\the\toks\tw\space\the\toks@}\fi
360   }\expandafter\rt@DB@List\fi}

```

`\useProbDBs{list}` Is an alias of `\useTheseDBs`.

```
361 \let\useProbDBs\useTheseDBs
```

`\viewDB{name}` Typeset the entire contents of a DB Test file. The argument *name* is the name of the DB Test file (as in `\BRTVToks{DB1}`, here DB1 is the *name*). The DB test files should be input using `\useProbDBs`.

```

362 \def\viewDB#1{\useRTName{#1}\rt@nCnt\z@
363   \edef\nSTOP{\@nameuse{nMax4}\rt@BaseName}}%
364   \loop\advance\rt@nCnt\@ne
365     \rtTokByNum{\the\rt@nCnt}%
366   \ifnum\rt@nCnt<\nSTOP\repeat
367 }

```

`\getR@nIndx` The macro `\getR@nIndx` executes with each entry of `\@nameuse{#1-List}`. For an index value of `\Indx`, the macro goes through the arguments to the `\Indx`'th argument and reads the value of the argument at that point. It returns the argument of the `\Indx`'th as `\ranIndx`; eg, if `\Indx=1`, then `\ranIndx=3`, for the above example.

```

368 %% uses \@tempcnta and \Indx
369 \def\getR@nIndx#1{\def\argi{#1}%
370   \ifx\argi\rt@STOP
371     % no match, something is wrong
372     \edef\ranIndex{-1}\else
373     \advance\@tempcnta\@ne
374     \ifnum\Indx=\@tempcnta
375       \def\ranIndx{#1}\fi
376   \fi
377 }

```

`\xdb@unique{name}` An add-on command to `\useRanTok`. The command attempts to create a unique choice of a problem over several versions of the same document. This may not be possible if there are not enough choices to satisfy the number of declared versions.

The `\xdb@unique` seems to work when the eqexam document has multiple parts (more than one exam environments). The id files for the parts are all combined;

ideally, for multiple part exams, the second part draws from a set of DB test files different from the first part, as long as there are enough problems to choose from.

Note that, if there is not a unique choice for a question from the designated DB test file, `\xdb@unique` reverts to the original choice so there may be duplicates across versions of the document.

```
378 \def\rt@NoAltChoice#1#2{\PackageWarning{ran_toks}
379 {Cannot find an alternative to #1-#2,\MessageBreak
380 will use it but it may be a duplicate\MessageBreak
381 question}}
382 \def\xdb@unique#1{\@tempcnta\z@
383 \def\rt@STOP{\relax}%
```

We use the randomized list for the $\langle name \rangle$

```
\@nameuse{#1-List} is the randomized list: eg,
\{3}\{2}\{4}\{5}\{1}
```

```
384 \let\\\relax\edef\x{\@nameuse{#1-List}}%
385 \toks@=\expandafter{\x}\let\\\getR@nIndx
386 \the\toks@\rt@STOP
```

We take as the default choice the original choice

```
387 \xdef\rt@next{\noexpand
388 \@nameuse{rtRanTok\rt@orig@Indx#1}}%
```

Begin to look at the results of `\the\toks@\rt@STOP`,

```
389 \ifnum\ranIndx>\m@ne
```

If `\ranIndx` is -1, we use the original index.

```
390 \edef\rt@orig@ranIndx{\ranIndx}%
391 \expandafter
392 \ifx\csname#1-\ranIndx\endcsname\relax
```

This question has not been chosen earlier, so we'll use it.

```
393 \xdef\rt@next{\noexpand
394 \@nameuse{rtRanTok\Indx#1}}%
395 \else
```

The question has been chosen in an earlier version of the document. Find the next higher unused one (cycle search).

```
396 \@tempcntb\z@
397 \rt@nCnt\rt@orig@Indx\relax
```

As we move into the `\@whilenum` loop, we take as the default the original index. The loop may overwrite the definition of `\rt@next`.

```
398 \xdef\rt@next{\noexpand\rt@NoAltChoice{#1}{\rt@orig@Indx}\noexpand
399 \@nameuse{rtRanTok\rt@orig@Indx#1}}%
400 \@whilenum\@tempcntb<\@nameuse{nMax4#1}\do{%
401 \advance\@tempcntb\@ne
402 \advance\rt@nCnt\@ne
```

If the count is at the `nMax4` value, we start over from the beginning.

```
403 \ifnum\rt@nCnt>\@nameuse{nMax4#1}\rt@nCnt\@ne\fi
```

We search through \toks@ again, so we have to initialize the dependent variables: \Indx, the index to search for; \@tempcnta the counter that is used by \getR@nIndx; nothing has changed \let\\getR@nIndx should still be in effect.

```

404     \edef\Indx{\the\rt@nCnt}\@tempcnta\z@
405     \the\toks@\\rt@STOP
    If \ranIndx is -1, we use the original index.
406     \ifnum\ranIndx>\m@ne
407     \expandafter
408     \ifx\csname#1-\ranIndx\endcsname\relax
409     \pkgNotifType{ran_toks}{#1-\rt@orig@ranIndx\space
410     has already been used,\MessageBreak
411     will use #1-\ranIndx}%
412     % exit the \@whilenum loop
413     \@tempcntb\@nameuse{nMax4#1}%
414     \advance\@tempcntb\@ne
415     \fi
416     \fi
417     \xdef\rt@next{\noexpand\@nameuse{rtRanTok\Indx#1}}%
418     }% do
419     \ifnum\@tempcntb=\@nameuse{nMax4#1}\relax
420     \xdef\rt@next{\noexpand
421     \rt@NoAltChoice{#1}{\rt@orig@ranIndx}\noexpand
422     \@nameuse{rtRanTok\rt@orig@Indx#1}}%
423     \fi
424     \fi
425 \fi
426 }

```

\uniqueXDBChoicesOn Here is the operational definition of \uniqueXDBChoicesOn; when executed in the preamble, an attempt is made to select only problems that have not already been chosen in any prior renditions of the same source file.

```

427 \renewcommand{\uniqueXDBChoicesOn}{\xDBUniquetrue}
428 \let\xdbunique\xdb@unique}

```

\InputUsedIDs Input the user ID (CUT) files. These are files that document which questions were used for the various versions of the exam.

```

429 \newif\ifrt@InputUsedIDs\rt@InputUsedIDsfalse
430 \def\InputUsedIDs{\rt@InputUsedIDstrue
431 \bgroup
432 \setcounter{eq@count}{0}%
433 \let\rt@InputUsedIDs\@empty
434 \let\rt@InputUsedIDsFIs\@empty
435 \@whilenum \value{eq@count}<\eq@nVersions\relax\do
436 {%
437 \stepcounter{eq@count}%
438 \g@addto@macro\rt@InputUsedIDs{\if\selVersion}%
439 \g@addto@macro\rt@InputUsedIDsFIs{\fi}%
440 \edef\x{\Alph{eq@count}}%
441 \edef\y{\noexpand\g@addto@macro\noexpand

```



```

442     \rt@InputUsedIDs{\x\expandafter\noexpand
443     \csname else\endcsname\noexpand\rt@IIFE}}\y
444     \edef\x{{\x}}\expandafter
445     \g@addto@macro\expandafter\rt@InputUsedIDs\expandafter{\x}%
446   }% do
447   \expandafter\g@addto@macro\expandafter
448   \rt@InputUsedIDs\expandafter{\rt@InputUsedIDsFIs}%
449   \egroup
450   \rt@InputUsedIDs
451   \AtBeginDocument{\rt@ABD}%
452 }
453 \@onlypreamble\InputUsedIDs

```

A convenience command used by \InputUsedIDs.

```

454 \def\rt@IIFE#1{\InputIfFileExists{\jobname-ver#1.cut}
455   {\PackageInfo{ran_toks}{Inputting \jobname-ver#1.cut}}
456   {\PackageInfo{ran_toks}{Cannot find \jobname-ver#1.cut}}}
457 </dbapp>
458 <*package>
459 </package>

```

6 Index

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

Symbols	F
<code>\%</code>	97
<code>\:</code>	341, 342
<code>\@bspback</code>	81
<code>\@let@token</code>	336, 342, 344
<code>\@makeother</code>	82
<code>\@onlypreamble</code>	59, 61, 453
<code>\@rgi</code>	187
<code>\@sptoken</code>	336
<code>\^</code>	83
A	
<code>\active</code>	83
<code>\Alph</code>	440
<code>\argi</code>	369, 370
<code>\AtBeginDocument</code>	451
<code>\AtEndDocument</code>	101
<code>\AtEndOfPackage</code>	44
B	
<code>\bRTVToks</code>	164, <u>182</u>
C	
<code>\copyRanToks</code>	<u>246</u>
<code>\CurrentOption</code>	51
D	
<code>\day</code>	8
<code>\DeclareOption</code>	38, 51
<code>\displ@yListRandomly</code>	312, 314
<code>\displayListRandomly</code>	<i>11</i> , 311
<code>\divide</code>	15, 146
<code>\dospecials</code>	82
E	
<code>\egroup</code>	308, 449
<code>\endrtVW</code>	212, 214
<code>\endrtVWi</code>	212
<code>\endrtVWii</code>	214
<code>\endverbatimwrite</code>	88, 207
environments:	
<code>rtVW</code>	<u>186</u>
<code>\eq@nVersions</code>	334, 435
<code>\eRTVToks</code>	165, <u>183</u>
F	
<code>\filename@area</code>	347
<code>\filename@base</code>	347
<code>\filename@ext</code>	347, 348
<code>\filename@parse</code>	345
<code>\first</code>	<i>12</i> , 320
<code>\futurelet</code>	342, 344
G	
<code>\g@addto@macro</code>	438, 439, 441, 445, 447
<code>\getR@nIndx</code>	<i>14</i> , 369, 385
I	
<code>\i</code>	<i>12</i>
<code>\ifFileExists</code>	63
<code>\ifhmode</code>	186
<code>\ifrt@InputUsedIDs</code>	34, 196, 429
<code>\ifrt@recording</code>	289, 302
<code>\ifrtdebug</code>	31, 115, 117, 129, 137, 142, 222
<code>\ifsaveseed</code>	33, 91
<code>\ifviewIDs</code>	191, 195
<code>\ifwerandomize</code>	32, 109
<code>\ifxDBUnique</code>	192, 196, 226, 332
<code>\Indx</code> .	296, 298, 300, 302, 303, 368, 374, 394, 404, 417
<code>\InitSeedValue</code>	12, 67, 71, 90, 98
<code>\input</code>	3, 225
<code>\InputIfFileExists</code>	39, 355, 454
<code>\inputlineno</code>	152, 161, 170
<code>\inputRandomSeed</code>	79
<code>\InputUsedIDs</code>	<u>429</u>
L	
<code>\last</code>	<i>12</i> , 320, 322
<code>\lastRandomNum</code>	68, 70
<code>\lessone</code>	<i>12</i> , 322
<code>\loop</code>	364
M	
<code>\m@ne</code>	121, 146, 259, 321, 389, 406
<code>\makeInfoAWarning</code>	<i>11</i> , 287
<code>\month</code>	9
<code>\multiply</code>	7–9, 17, 19, 20, 147
N	
<code>\NeedsTeXFormat</code>	48

`\newwrite` 35, 92, 190
`\nextrandom` 4, 11
`\nSTOP` 363, 366
`\nToksFor` 10, 153, 270

O

`\openin` 66
`\openout` 93, 199, 331
options:
 `usedbapp` 3

P

`\PackageInfo` 40, 41, 64, 73, 288, 455, 456
`\PackageWarning` 282, 287, 346, 378
`\PackageWarningNoLine` 150, 160, 169
`\PassOptionsToPackage` 51
`\pkgNotifType` 287, 288, 409
`\ProbDBWarningMsg` 345, 356
`\ProcessOptions` 45, 52
`\ProvidesPackage` 49

R

`\r@ndToks` 8, 178, 229, 230, 234
`\r@nToks` 176, 178, 224
`\r@nVToks` 8, 185, 215, 223
`\randomi` 5–10, 12, 14, 18, 19, 21, 22, 58, 70, 71, 90, 99
`\ranIndex` 372
`\ranIndx` 375, 389, 390, 392, 406, 408, 411
`\ranToks` 164, 174
`\ranToksOff` 3, 57
`\ranToksOn` 3, 56
`\read` 67, 68
`\readsavfile` 65–69
`\reorderRanToks` 241
`\repeat` 366
`\RequirePackage` 2, 53
`\reVerbEnd` 186, 208
`\rt@ABD` 332, 333, 451
`\rt@badIndex` 6, 150, 296
`\rt@badTokName` .. 6, 159, 243, 249, 272, 277, 292, 316
`\rt@BaseName` 112, 180, 181, 184, 185,
 200, 204, 216, 232, 239, 268, 269, 275, 290, 363
`\rt@DB@List` 350, 357, 359, 360
`\rt@dblList` 349, 351
`\rt@endToks` 9, 234, 238
`\rt@getRanNum` 30, 114, 115, 134
`\rt@gettonil` 335, 338
`\rt@ifspc` 336, 342, 344
`\rt@IIFE` 443, 454
`\rt@InputUsedIDs` 433, 438, 442, 445, 448, 450

`\rt@InputUsedIDsfalse` 34, 429
`\rt@InputUsedIDsFIs` 434, 439, 448
`\rt@InputUsedIDstrue` 430
`\rt@inputVerb` 8, 221, 225
`\rt@IWVO` 89, 98, 99, 202, 203
`\rt@listIn` 25, 102,
 104, 105, 107, 111, 117, 118, 124, 218, 221–224
`\rt@listInHold` 104, 105, 140, 141, 218, 220, 222
`\rt@listOut`
 .. 27, 107, 111, 112, 128, 130, 135, 136, 138, 237
`\rt@listOutHold` 135, 136
`\rt@loopTest` 119, 121
`\rt@mod` 149, 155, 296
`\rt@modarith` 6, 145, 295
`\rt@msgi` 95, 98
`\rt@msgii` 96, 99
`\rt@namedef` 37, 112, 184, 239, 254, 257, 268
`\rt@nCnt` 29, 102–104, 116, 133, 134,
 182, 184, 198, 200, 204, 218, 219, 221, 236,
 255–259, 267, 268, 362, 364–366, 397, 402–404
`\rt@newListIn` 26, 107, 124, 125, 140–142
`\rt@next` 123, 127, 131, 230, 233, 234,
 292, 303, 309, 337, 338, 387, 393, 398, 417, 420
`\rt@nextDB` 354, 358
`\rt@NIL` 178, 179, 233
`\rt@nMax` 28, 108,
 114, 121, 122, 178, 217, 219, 231, 232, 239, 240
`\rt@NoAltChoice` 378, 398, 421
`\rt@OpenProbIds` 330, 332, 334
`\rt@orig@Indx` 300, 388, 397–399, 422
`\rt@orig@ranIndx` 390, 409, 421
`\rt@PAR` 228, 230
`\rt@performRanDefns` 9, 235, 240, 244, 260
`\rt@PkgInpt` 38, 39
`\rt@populateList` 5, 102, 108
`\rt@post` 319, 324
`\rt@prior` 312, 324
`\rt@processi` 113, 133
`\rt@RandomizeList` 5, 106, 236
`\rt@randomizeList` 5, 110, 113, 125
`\rt@recordAsUsed` 225–227, 302
`\rt@recordingfalse` 318
`\rt@recordingtrue` 289
`\rt@rgi` 229, 230, 316
`\rt@ssign` 10, 237, 267
`\rt@STOP` 370, 383, 386, 405
`\rt@strpspcs` 344, 353
`\rt@useLastAsSeed` 60, 62
`\rt@Verb@write` 35, 199, 201, 208, 209
`\rt@warnTokName` 168, 262

<code>\rt@writeSeedData</code>	91, 101	<code>usedbapp</code> (option)	3
<code>\rt@xifspc</code>	337, 341	<code>\useLastAsSeed</code>	4, 60, 61, 79
<code>\rtcsarg</code>	36, 194, 226, 227	<code>\useProbDBs</code>	361
<code>\rtdebugfalse</code>	31	<code>\useRandomSeed</code>	80
<code>\rtPkgInpt</code>	38, 43, 44	<code>\useRanTok</code>	10, 151, 155, 290, 324
<code>\rtTokByNum</code>	10, 275, 365	<code>\useRTName</code>	175, 180, 182, 241, 253, 319, 362
<code>\rtVW</code>	211, 213	<code>\useTheseDBs</code>	345, 361
<code>rtVW</code> (environment)	186	<code>\useThisSeed</code>	3, 58, 59, 80
<code>\RTVWHook</code>	188, 202		
<code>\rtVWHook</code>	187, 189	V	
<code>\rtVWi</code>	211	<code>\verbatim@line</code>	86
<code>\rtVWId</code>	195, 204	<code>\verbatim@out</code>	85, 89, 94, 201
<code>\rtVWii</code>	213	<code>\verbatim@processline</code>	84
		<code>\verbatim@start</code>	87
S		<code>\verbatimwrite</code>	81, 205
<code>\saveseedfalse</code>	58	<code>\viewDB</code>	362
<code>\saveseedinfo</code>	92–94, 100	<code>\viewIDsfalse</code>	191
<code>\saveseedtrue</code>	33		
<code>\selVersion</code>	331, 438	W	
<code>\setcounter</code>	432	<code>\werandomizefalse</code>	57
<code>\setrannum</code>	114	<code>\werandomizetrue</code>	32, 56
<code>\stepcounter</code>	437	<code>\write</code>	85, 89, 193, 208
		<code>\wrtProbIds</code>	193, 196
T		<code>\wrtprobids</code>	190, 193, 331
<code>\temp@expand</code>	351, 352		
<code>\textbf</code>	272, 277, 316	X	
<code>\texttt</code>	284	<code>\x</code>	384, 385, 440, 442, 444, 445
<code>\time</code>	6	<code>\xdb@unique</code>	14, 305, 382, 428
<code>\to@nilarg</code>	335, 353	<code>\xdbunique</code>	285, 286, 301, 428
<code>\toks</code>	357, 359	<code>\xDBUniquefalse</code>	192, 281, 285
		<code>\xDBUniquetrue</code>	427
U			
<code>\uccode</code>	97	Y	
<code>\uniqueXDBChoicesOff</code>	11, 285	<code>\y</code>	441, 443
<code>\uniqueXDBChoicesOn</code>	11, 16, 281, 283, 427	<code>\year</code>	7
<code>\uppercase</code>	97		

7 Change History

v1.0b (2013/07/29)	Fixed a bug, when the first two tokens #1 are the same, we get an incorrect decision	8
General: Added <code>\displayListRandomly</code>		11
v1.0c (2013/08/03)	General: Save the initial seed value to <code>\rtInitSeedValue.</code>	2
v1.0d (2013/08/03)	General: Added conditional input of <code>random.tex</code> ..	2
v1.0e (2016/02/06)	General: Added optional argument to <code>\displayListRandomly</code>	11
v1.1 (2017/05/04)	General: Added second optional argument to <code>\displayListRandomly</code>	11
General: Added second optional argument to <code>\displayListRandomly</code>		5
v1.2 (2019/12/28)	General: Added dummy package <code>ran-toks</code>	3
General: Added dummy package <code>ran-toks</code>		7
<code>rtVW</code> : Defined <code>\rtVWHook</code>		7

v1.3 (2021/01/14)		v1.3.3 (2021/05/29)	
General: Added <code>usedbapp</code> option	3	General: In support of nested	
Added several commands and macro to		<code>\bRTVToks/\eRTVToks</code> constructs, replace the	
continue to support a DB application.	13	counter <code>\rt@nCnt</code> with <code>\@tempcntb</code>	11
v1.3.1 (2021/01/19)		<code>rtVW</code> : Assign <code>\rtVWi</code> and <code>\endrtVWi</code>	8
General: Added <code>\ifrt@recording</code>	11	v1.4 (2021/06/06)	
v1.3.2 (2021/05/25)		General: Rounded up v1.3.3 to v1.4 for	
General: Insert optional argument for		publication	2
<code>\useRanTok</code> within <code>\displayListRandomly</code> .	12		