The **Permute** Package

Copyright 1997–1999 by Carsten Heinz

Version 0.12

Abstract

The `permute` package inputs, outputs and composes permutations. For example, \( \pmt{(123)} \circ \pmt{(321)} = \pmt{(123)(321)} \) produces (123) \( \circ \) (132) = id. A misleading example is \( (a\ldots z) \circ (z\ldots a) = \pmt{(a\ldots z)(z\ldots a)} \) — misleading since the package doesn’t care about the human interpretation of “…”.

1 User’s guide

1.1 Software license

`permute.dtx` and `permute.ins` and all files generated from these files are referred to as ‘the permute package’. It is distributed under the terms of the L\TeX\ Project Public License from CTAN archives in directory `macros/latex/base/lppl.txt`. Either version 1.0 or, at your option, any later version. The use of the package is completely free.

Permission is granted to modify the `permute` package. You are not allowed to distribute any changed version of the package, neither under the same name nor under a different one.

Send comments, ideas and bug reports via electronic mail to `cheinz@gmx.de`.

1.2 Installation

1. Following the T\TeX\ directory structure (TDS) you should put the files of the package into directories as follows:

   ```
   \texttt{permute.dvi} \rightarrow \texttt{texmf/doc/latex/permute}
   \texttt{permute.dtx, permute.ins} \rightarrow \texttt{texmf/source/latex/permute}
   ```

   Of course, you need not to use the TDS. Simply adjust the directories below.

2. Create the directory `texmf/tex/latex/permute`.

3. Change the working directory to `texmf/source/latex/permute` and run `permute.ins` through T\TeX.
4. Move the generated file to `texmf/tex/latex/permute` if this is not already done.

5. If your TeX implementation uses a filename database, update it.

1.3 Input and output formats

General notation. First we restrict ourselves to $S_1, \ldots, S_9$:

$$S_n := \{ f : \{ 1, \ldots, n \} \to \{ 1, \ldots, n \} \mid f \text{ is one-to-one and onto} \}.$$  

A permutation $f \in S_n$ can be written **verbose** as an explicit sequence of pre-image/image pairs like this:

$$f = \left( \begin{array}{cccc} 1 & 2 & \cdots & n \\ f(1) & f(2) & \cdots & f(n) \end{array} \right).$$

A permutation $\sigma$ is a **cycle** and written $\sigma = (x_1 x_2 \ldots x_k)$ if and only if there are distinct numbers $x_1, x_2, \ldots, x_k \in \{ 1, \ldots, n \}$ satisfying

$$
\begin{align*}
\sigma(x_i) &= x_{i+1} & \text{for all } 1 \leq i < k \\
\sigma(x_k) &= x_1 \\
\sigma(x) &= x & \text{for all } x \in \{ 1, \ldots, n \} \setminus \{ x_1, \ldots, x_k \}.
\end{align*}
$$

This means that a cycle $(x_1 x_2 \ldots x_k)$ maps $x_1$ to $x_2$, $x_2$ to $x_3$, ...., $x_k$ back to $x_1$, and fixes all other elements. Each permutation in $S_n$ can be written as a composition of cycles, for example $(12)(354) = (12)(354)$—we leave out the “◦”. Note that $S_n$ is not commutative in general and that we compose from the right to the left. In the example 3 is mapped to 5 and then 5 stays 5 since unchanged by (12).

Input formats. If you want to enter a permutation cycle based, just write the cycles after each other: $(12)(354)$ would be legal; there must not be a `\circ` in between. The verbose input format lists all pre-image/image pairs without any separators, but a space is allowed in between. The permutation above could also be entered as `12 21 35 43 54`. The package distinguishes the two formats by looking at the first token: If and only if it’s a left parenthesis, the package accepts cycles.

Now we drop the restriction $n \leq 9$ and the limitation of permuting numbers; we can do it with nearly arbitrary (token) strings. To enter a string as pre-image, image or inside a cycle, enclose the string in braces. That’s all! For example, I typed

```latex
1{f(1)} 2{f(2)} \ldots n{f(n)}
```

for $\left( \begin{array}{cccc} 1 & 2 & \cdots & n \\ f(1) & f(2) & \cdots & f(n) \end{array} \right)$, which actually isn’t a permutation at all.

In the sequel `<pmt>` means either a verbose sequence or a sequence of cycles. **Do not use \empty or \relax or equivalent definitions inside `<pmt>`**.
Output formats. The package provides a cycle based and two verbose output formats. Some of the commands described below have an optional ⟨print order⟩ argument. Now let ⟨print order⟩ equal \(n_1n_2\ldots n_k\) where each \(n_i\) is a single token or a braced string. If \(n_i\) and \(n_{i+1}\) both appear in a permutation and appear in different cycles, then the cycle containing \(n_i\) is printed first. Moreover the cycle starts with the element \(n_i\). Cycles not covered by ⟨print order⟩ are printed as they appear in the internal data format. Some examples on printing \((12)(34)(56)\):

<table>
<thead>
<tr>
<th>⟨print order⟩</th>
<th>results in</th>
<th>⟨print order⟩</th>
<th>results in</th>
</tr>
</thead>
<tbody>
<tr>
<td>no order</td>
<td>((12)(34)(56))</td>
<td>empty</td>
<td>((56)(34)(12))</td>
</tr>
<tr>
<td>2</td>
<td>((21)(56)(34))</td>
<td>23</td>
<td>((21)(34)(56))</td>
</tr>
<tr>
<td>6</td>
<td>((65)(34)(12))</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

All results represent the same permutation. Note the difference between ‘no order’ and ‘empty’: The package uses a standard order if you don’t request a special one.

There is some danger if you want to use a single token string as ⟨print order⟩. In this case you must enclose the string in two level of braces. Use ⟨print order⟩=\{{one}\} to control \((\{one\}\{two\})(\{two\}\{three\})(\{three\}\{four\})\), for example. Since \TeX{} discards one group level, \{one\} would lead to the order \(o, n, e\). However, ⟨print order⟩=\{{one}\{two\}} needs no extra braces.

For the verbose output formats, ⟨print order⟩ plays the role of domain. The package uses exactly the elements and order, i.e. all pre-images not appearing in ⟨print order⟩ are not printed, and we assume image=pre-image if pre-image appears in ⟨print order⟩ but not in the permutation. Some examples on printing \(1\{f(1)\} 2\{f(2)\} \ldots n\{f(n)\}\):

<table>
<thead>
<tr>
<th>⟨print order⟩</th>
<th>results in</th>
</tr>
</thead>
<tbody>
<tr>
<td>no order</td>
<td>[\begin{array}{</td>
</tr>
<tr>
<td>n\ldots 21</td>
<td>[\begin{array}{</td>
</tr>
<tr>
<td>2\ldots n{n+1}</td>
<td>[\begin{array}{</td>
</tr>
<tr>
<td>empty</td>
<td>()</td>
</tr>
</tbody>
</table>

The first example doesn’t print ‘\(\ldots\)’ and ‘\(f(n)\)’ since the pre-images ‘\(\ldots\)’ and ‘\(n\)’ don’t appear in the standard printing order (which is the domain here). But it shows image=pre-image pairs not in the permutation since the standard domain defines the elements as pre-images.

Note: (a) The (full) verbose format uses math mode and the \TeX{}-primitive \atop. The latter causes a warning if used together with amsmath.sty. (b) The package defines a short verbose output format, too. It prints the row of images only. Don’t take it for the cycle based format!

Finally, some commands also have a star-form which separate the output like this: \((1234)\). It is useful if you use strings instead of numbers, for example the permutation (one two three four) is printed with a *-command.
1.4 User commands

\texttt{\textbackslash pmt}\langle (\textit{print\ order}) \rangle \{\langle \textit{pmt} \rangle \}

calculates the composition of the cycles (if any) and prints the permutation:
\texttt{\pmt\{12\}(23)(34)} prints (1234) and \texttt{\pmt\{12\ 23\ 34\ 41\}} gives (1234).
Note that this command outputs cycles only. Some examples:

\begin{itemize}
  \item \texttt{\pmt\{12\}(34)(56)} prints (12)(34)(56)
  \item \texttt{\pmt\{12\}(34)(56)} prints (2\ 1)(5\ 6)(3\ 4)
  \item \texttt{\pmt\{12\}(34)(56)} prints (21)(34)(56)
  \item \texttt{\pmt\{12\}(34)(56)} prints (6\ 5)(3\ 4)(1\ 2)
\end{itemize}

The macro \texttt{\textbackslash pmt\textbackslash print\ order} contains the standard printing order, see 1.5.

\texttt{\textbackslash pmtv}\langle (\textit{print\ order}) \rangle \{\langle \textit{pmt} \rangle \}

prints a verbose form of the permutation. The commands \texttt{\textbackslash pmtv\textbackslash shorttrue} and \texttt{\textbackslash pmtv\textbackslash shortfalse} controls whether the package prints only the row of images or the full pre-image/image format.

\texttt{\textbackslash pmt\textbackslash table}\langle (\textit{print\ order}) \rangle \{\langle \textit{list\ of\ pmts} \rangle \} \{\langle \textit{list\ of\ pmts} \rangle \}

\texttt{\textbackslash pmtv\textbackslash table}\langle (\textit{print\ order}) \rangle \{\langle \textit{list\ of\ pmts} \rangle \} \{\langle \textit{list\ of\ pmts} \rangle \}

The commands compose each $\sigma_1$ of the first list with each $\sigma_2$ of the second list and write the result $\sigma_1 \circ \sigma_2$ in row $\sigma_1$ and column $\sigma_2$. For example,

\begin{verbatim}
$$\texttt{\pmt\table\{(),(12),(13),(23),(123),(132)\}\{(),(23),(123),(132)\}}$$
\end{verbatim}

creates

\begin{center}
\begin{tabular}{c|cccc}
  $\circ$ & id & (12) & (13) & (23) & (123) & (132) \\
  \hline
  id & id & (12) & (13) & (23) & (123) & (132) \\
  (12) & (12) & id & (132) & (123) & (23) & (13) \\
  (13) & (13) & (123) & id & (132) & (12) & (23) \\
  (23) & (23) & (132) & (123) & id & (13) & (12) \\
  (123) & (123) & (13) & (23) & (12) & (132) & id \\
  (132) & (132) & (23) & (12) & (13) & id & (123) \\
\end{tabular}
\end{center}

'(()) stands for the identity map here. \textit{Do not write} \texttt{\pmt\table\{(),(12),...}}. You may write \texttt{\pmt\table\{12\),(123)\} \{(),(23),(123),(132)\} to typeset a piece of the table or use \texttt{\pmtv\table} to print all permutations verbose. The optional arguments effect all printed permutations.
If you create really big tables like the one of $S_5$, you surely want to cut the whole table in pieces and produce subtables on different pages. This leads to alignment problems since the first column on the first page need not to have the width of the first column on the second page. Bad luck!

Now we discuss how to calculate with the \texttt{permute} package. Let $\langle \text{current pmt} \rangle$ denote the (internal) current permutation and $\langle \text{name} \rangle$ another internal (stored) permutation.

\begin{verbatim}
\pmtload{\langle \text{name} \rangle}
\pmtsave{\langle \text{name} \rangle}
\pmtid{\langle \text{name} \rangle}
\pmtdo{\langle \text{name} \rangle}{\langle \text{pmt} \rangle}
\pmtcirc{\langle \text{name} \rangle}{\langle \text{pmt} \rangle}
\pmtprint{*}{\langle \text{print order} \rangle}
\pmtvprint{*}{\langle \text{print order} \rangle}
\pmtimageof{\langle \text{name} \rangle}{\langle \text{pre-image} \rangle}
\pmtpreimageof{\langle \text{name} \rangle}{\langle \text{image} \rangle}
\end{verbatim}

If you use any optional $\langle \text{name} \rangle$, this permutation replaces $\langle \text{current pmt} \rangle$. For example, $\texttt{\pmtid[a]}$ makes the permutation $a$ to be the identity map or $\texttt{\pmtdo[a]}{\langle \text{pmt} \rangle}$ performs $a ← \langle \text{pmt} \rangle \circ a$.

Finally examples which all print the result of $(f_1 \ldots f_5) \circ (f_5 f_4 \ldots f_1)$, namely \[
\begin{pmatrix}
\begin{array}{cccc}
1 & \ldots & k & l \\
f_1(f_1) & \ldots & f(l) & f(k) & \ldots & f(n)
\end{array}
\end{pmatrix}.
\]

\begin{verbatim}
\pmtid
\pmtdo{f_1 \ldots f_5} k{f(k)} l{f(l)} \ldots n{f(n)}
\pmtcirc{(k l)}
\pmtvprint{1 \ldots k l \ldots n}
\pmtid
\pmtdo{(k l)}
\pmtdo{f_1 \ldots f_5} k{f(k)} l{f(l)} n{f(n)}
\pmtvprint{1 \ldots k l \ldots n}
\end{verbatim}

We can drop the two \texttt{\ldots} pairs (also in the first example) since image and pre-image are equal.
1.5 Parameters

\pmtprintorder
contains the standard printing order which is used whenever you leave out or forget the optional \textit{(print order)} argument. By default it contains the sequence 123456789abcdefghi. You may adjust it to your needs, for example, \renewcommand*{\pmtprintorder{123456}} if you work with S6. This is especially good if you use the verbose printing format. You can forget the optional \textit{(print order)} in this case:

\renewcommand*{\pmtprintorder{123456}}
\pmtv{()}\quad \text{prints } \{1\ 2\ 3\ 4\ 5\ 6\}
\pmtv{(123)}\quad \text{prints } \{1\ 2\ 3\ 4\ 5\ 6\}
\pmtv{(12)(23)(16)(34)(45)}\quad \text{prints } \{1\ 2\ 3\ 4\ 5\ 6\}

\pmtseparator
contains the separator used for the optional star. By default it is a space: \newcommand*{\pmtseparator{ }}. Since it’s not a backslashed space \textbackslash{} it is ignored in math mode, in particular in the verbose output format. But you may write \renewcommand*{\pmtseparator{\textbackslash{}}}.

\pmtidname
contains the string which is printed in the cycle based format if a permutation is the identity map. It is predefined via \newcommand*{\pmtidname{id}}.

\pmtldelim \pmtrdelim
contain the left respectively right delimiter for the verbose format. \texttt{\left(} and \texttt{\right)} are used by default.

\pmttableborders
contains \texttt{lt} which makes a left border column and a top border line. You may redefine it to be empty or to contain \texttt{l}, \texttt{t}, \texttt{lt} or \texttt{tl}.

\pmtarraystretch
contains 2 and is used as \texttt{arraystretch} for the table. You may write \renewcommand*{\pmtarraystretch{1.3}} and get more compact tables:

<table>
<thead>
<tr>
<th></th>
<th>id</th>
<th>(12)</th>
<th>(23)</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>id</td>
<td>(12)</td>
<td>(23)</td>
</tr>
<tr>
<td>(12)</td>
<td>(12)</td>
<td>id</td>
<td>(123)</td>
</tr>
<tr>
<td>(23)</td>
<td>(23)</td>
<td>(132)</td>
<td>id</td>
</tr>
</tbody>
</table>