

コンピュータ概論のレポート

— TeX マクロ —

出題 2011/12/13

学科 数学科

学年 1 年

番号 9999

氏名 栗野俊一

1 マクロの利用例

色々な人が色々なマクロを定義して公開しているので、それを利用するには、そのマクロが定義されているファイルを取り込めばよい。

- `\input` で取り込む。

次の例は、`macro.tex` の中に定義されているマクロを利用する場合に使う。`macro.tex` はタイプセットしたい `TeX` ファイルと同じフォルダにおいておく。

`\input` マクロを利用 “`macro.tex`” を利用する

```
\input{macro.tex}
```

- `\usepackage` で取り込む。

次の例は、“`ascmac.sty`” の中に定義されているマクロを利用する場合に使う。“`ascmac.sty`” は、予めインストール済 (特定のフォルダ¹においてある..) とする。

タイプセットしたい `TeX` ファイルと同じフォルダにおいてもよい。

`\usepackage` マクロを利用して “`ascmac.sty`” を取り込む

```
\usepackage{ascmac}
```

2 マクロの定義

2.1 マクロの定義の位置

マクロの定義は、どこでもできるが、定義された行から後ろで有効になるので、できるだけ、先頭で行う事が望ましい。

その一方、マクロの定義の中で他のマクロを利用する事が多いのでその定義の後に後になる。

したがって、通常自分が定義するマクロは、`\usepackage` の後、`\begin{document}` の前におくことが基本となる。

`\input` などは、`\usepackage` の直後にすると良いかもしれない。

2.2 単純マクロの定義

マクロを定義する場合は、「マクロを定義するマクロ (`\newcommand`, `\renewcommand` など..)」を利用するとよい。

マクロ名はなんでもよいが、既存のマクロと名前が被るといけないので、自分が作るマクロ名は、大文字で始めるとか、`My` や `Kurino` などの、専用のキーワードを先行させるなどの工夫² をするとよい。

- `\newcommand` で定義する。

¹`C:\w32tex\share\texmf\platex\base` にある。

²`style` ファイル内では実は、特別な工夫 (`\makeatletter`, `\makeatother`) をする事によって、名前の中に特別な文字 (@) を含め、重複をさけるようになっている。

定義 単純なマクロ (`\MyMacro`) の定義

`\newcommand` による単純なマクロ定義
`\newcommand{\MyMacro}{マクロ定義の例}`

参照 マクロの利用例

マクロの利用例
文中に「`\MyMacro`」とマクロ名を記述するだけ。
マクロ名の後ろには空白をいれよう。

結果 マクロの利用結果

マクロの利用結果
文中に「マクロ定義の例」とマクロ名を記述するだけ。マクロ名の後ろには空白をいれよう。

- `\def`³ で定義する。

定義 単純なマクロ (`\MyMaclo`) の定義

`\def` による単純なマクロ定義
`\def\MyMaclo{マクロ定義の別の例}`

参照 マクロの利用例

マクロの利用例
どちらで定義されていても、
文中に「`\MyMaclo`」とマクロ名を記述する事は同じ。

結果 マクロの利用結果

マクロの利用結果
どちらで定義されていても、文中に「マクロ定義の別の例」とマクロ名を記述する事は同じ。

2.3 自分の名前のロゴを作ってみる

TeX の特徴的なロゴ (`\TeX`) も、マクロの例になっており、文中で `\TeX` と書く事によって利用できる [1]。

このマクロは文字の表示位置を変更するだけで実現されている。

定義 単純マクロ `\MyName` を定義

³LaTeX の多くの命令は、TeX のマクロで実現されている。TeX には、基本命令があり、LaTeX の命令 (マクロの形で定義されている..) も最終的には、それを実現されている。`\def` は TeX の基本命令の一つであり、これがマクロ定義機能の基本となるし、また、LaTeX でも、それが利用できる。しかし、LaTeX では、表現の統一や、他の命令との兼ね合いから、`\newcommand` とういう新しい定義マクロが用意されており、以下では、そちらだけを説明する。

- マクロ定義が、みやすいように、適度に改行をいれている。
- ただ、マクロの定義内容の中に、無駄な改行の挿入をさけるために、文字の直後に%を入れて改行している。
- `\lower` で少し下げる。 `ex` は文字の高さを表わし、係数の `0.2` は、そのサイズの `0.2` 倍の意味となる。
- 係数に `-` をつければ逆方向の意味となる

自分の名前のロゴの定義

```
\newcommand{\MyName}{%
K%
\kern -.1em\lower .2ex\hbox{u}%
\kern -.1em\lower -.2ex\hbox{r}%
\lower -.4ex\hbox{i}%
\lower -.2ex\hbox{n}%
o}
```

参照 `\MyName` の参照

ロゴの利用例

私の名前は `\MyName` です。

結果 `\MyName` の表示結果

ロゴの表示

私の名前は K_ur_ino です。

2.4 引数付きマクロの定義

マクロ名の後ろに引数を付加する事により、より柔軟なマクロを作成することができる。

- `\newcommand` でマクロを定義する場合に、引数の個数を指定できる。
- マクロ定義の本体の中で `# + 数値 (n)` で、`n` 番目の引数を参照できる。
- 引数付きマクロを利用する場合は、マクロ名の後ろに引数を並べるが、その場合には、カーリーブラケット (「`{`」と「`}`」) で囲むのが望ましい。

定義 一つの引数付きマクロ `\Atenten` を定義する。

- 引数の個数が `[1]` という形で、一個ある事が示される。
- 引数の参照には `#` を使う。

引数付きマクロ `\Atenten` の定義

```
\newcommand{\Atenten}[1]{a_1, a_2, ..., a_{#1}}
```

参照 マクロ\Atenten の参照

- マクロの引数は、マクロ名の後に「{」と「}」で囲んで指定する。

\Atenten の利用方法

二つの数列 $\$ \backslash \text{Atenten}\{n\}$ と $\$ \backslash \text{Atenten}\{m\}$ を比較する事を考える。

結果 マクロ\Atenten の表示結果

\Atenten の表示結果

二つの数列 a_1, a_2, \dots, a_n と a_1, a_2, \dots, a_m を比較する事を考える。

マクロ \Atenten は、 a 数列だけなので、より汎用な \Tenten を定義する。

定義 二つの引数をもつマクロ\Tenten を定義する。

- 引数の個数が [2] という形で、二個ある事が示される。

引数付きマクロ \Tenten の定義

```
\newcommand{\Tenten}[2]{#1_1, #1_2, \cdots, #1_{#2}}
```

参照 マクロ\Tenten の参照

\Tenten の利用方法 1

二つの数列 $\$ \backslash \text{Tenten}\{b\}\{n\}$ と $\$ \backslash \text{Tenten}\{b\}\{m\}$ を比較する事を考える。

結果 マクロ\Tenten の表示結果

\Tenten の表示結果

二つの数列 b_1, b_2, \dots, b_n と b_1, b_2, \dots, b_m を比較する事を考える。

さらに、これを利用して、\Btenten を定義すると便利。

定義 マクロ\Btenten を定義する。

- \Btenten の定義の中で\Tenten を利用する。

引数付きマクロ \Btenten の定義

```
\newcommand{\Btenten}[1]{\Tenten\{b\}\{#1\}}
```

参照 マクロ\Btenten の参照

\Tenten の利用方法 2 (\Btenten)

二つの数列 $\$ \backslash \text{Btenten}\{n\}$ と $\$ \backslash \text{Btenten}\{m\}$ を比較する事を考える。

結果 マクロ`\Btnten` の表示結果

`\Btnten` の表示結果
二つの数列 b_1, b_2, \dots, b_n と b_1, b_2, \dots, b_m を比較する事を考える。

`\Atnten` も、`\Tnten` を利用するようにしてみる。

定義 マクロ`\Atnten` を「再」定義する。

- 再定義する場合は、`\renewcommand` を使う。

引数付きマクロ `\Atnten` の再定義
`\renewcommand{\Atnten}[1]{\Tnten{A}{#1}}`

参照 マクロ`\Atnten` の参照 (再定義された後に影響される)

`\Tnten` の利用方法 3 (`\Atnten`)
二つの数列 `$_Atnten{n}$` と `$_Atnten{m}$` を比較する事を考える。

結果 マクロ`\Atnten` の表示結果 (再定義された後から表示が変化する)

`\Atnten` の表示結果
二つの数列 A_1, A_2, \dots, A_n と A_1, A_2, \dots, A_m を比較する事を考える。

3 `\makeatletter` と `\makeatother`

通常、 $\text{T}_{\text{E}}\text{X}$ では、「@ (アットマーク)」は、英字とは認識されない。したがって、マクロ名の中でも利用できない。

ところが、`\makeatletter` 命令により、次に、`\makeatother` 命令がくるまでの間、@ も英字とみなすように $\text{T}_{\text{E}}\text{X}$ に指示を与える事ができる。

したがって、@ をマクロ名として含むようなマクロは、通常の利用者では定義も利用もできないため、マクロ名の重複という問題点を回避できる。

しかし、以下の議論では、参考文献 (「 $\text{LaTeX}2\text{e}$ まくろの八衢」[1]) にしたがって、@ を含むマクロを利用する事にする。

4 繰り返し

`\@for` を利用する事によって、「,(カンマ)」で区切られた文字列のリストを順に処理できる。

定義 マクロ`\@for` を利用した繰り返し

- `\@for` には「@」が含まれているので、これを利用するには、`\makeatletter`、`\makeatother` が必要になる事に注意。

引数付きマクロ `\DoFruit` の定義

```
\makeatletter
\newcommand{\DoFruit}[1]{%
  \@for\member:=#1\do{(\member)}%
}
\makeatother
```

参照 マクロ `\DoFruit` の参照

`\@for` の利用方法 (`\DoFruit`)

```
\DoFruit{orange,apple,banana}
```

結果 マクロ `\DoFruit` の表示結果

`\DoFruit` の表示結果

```
(orange)(apple)(banana)
```

参考文献

- [1] 藤田眞作, "「LaTeX2e まくろの八衢」オンライン版", <http://xymtex.com/fujitas/fujita.html>, 2004