

1

```

\count <register> = <number>      \count <register>
\dimen <register> = <dimen>        \dimen <register>
\skip <register> = <glue>          \skip <register>
\muskip <register> = <muglue>      \muskip <register>
\toks <register> = <token variable> \toks <register>
\toks <register> = { <token list> }

```

The first six commands listed here assign something to a register. The =’s in the assignments are optional. The remaining five control sequences are not true commands because they can only appear as part of an argument. They yield the contents of the specified register. Although you can’t use these control sequences by themselves as commands in text, you can use \the to convert them to text so that you can typeset their values.

You can name and reserve registers with the \newcount command and its relatives (p. ‘\@newcount’). Using these commands is a safe way to obtain registers that are known not to have any conflicting usage.

A \count register contains an integer, which can be either positive or negative. Integers can be as large as you’re ever likely to need them to be.¹ T_EX uses count registers 0–9 to keep track of the page number (see page 119 of *The T_EXbook*). \count255 is the only count register available for use without a reservation.

Example:

```
\count255 = 17 \number\count255
```

produces:

17

A \dimen register contains a dimension. Registers \dimen0 through \dimen9 and \dimen255 are available for scratch use.

Example:

```

\dimen0 = 2.5in
\hbox to \dimen0{ $\Leftarrow$ \hfil $\Rightarrow$ }

```

produces:

\Leftarrow

 \Rightarrow 3 in

A \skip register contains the dimensions of glue. Unlike a \dimen register, it records an amount of shrink and stretch as well as a natural size. Registers \skip0 through \skip9 and \skip255 are available for use without a reservation.


¹ Here’s the only exercise in this book: find out what’s the largest integer that T_EX will accept.

2

\ §0

Example:

```
\skip2 = 2in
$\Rightarrow$\hskip \skip2 $\Leftarrow$
```

produces:


A `\muskip` register is like a `\skip` register, but the glue in it is always measured in `\mu` (see “mathematical unit”, p. ‘`mathematical+unit`’). The size of a `\mu` depends on the current font. For example, it’s usually a little smaller in a subscript than in ordinary text. Registers `\muskip0` through `\muskip9` and `\muskip255` are available for use without a reservation.

Example:

```
\muskip0 = 24mu % An em and a half, no stretch or shrink.
$\mathop{a \mskip\muskip0 b}\limits^{\sim}\{a \mskip\muskip0 b\}$
% Note the difference in spacing.
```

produces:

$$\begin{array}{cc} a & b \\ a & b \end{array}$$

You can assign either a token variable (a register or a parameter) or a token list to a `\toks` register. When you assign a token list to a token register, the tokens in the token list are *not* expanded.

Once the tokens in a token list have been inserted into text using `\the`, they are expanded just like tokens that were read in directly. They have the category codes that they received when T_EX first saw them in the input.

Example:

```
\toks0 = {the \oystereaters\ were at the seashore}
% This assignment doesn't expand \oystereaters.
\def\oystereaters{Walrus and Carpenter}
\toks1 = \toks0
% the same tokens are now in \toks0 and \toks1
Alice inquired as to whether \the\toks1.
```

produces:

Alice inquired as to whether the Walrus and Carpenter were at the seashore.