This is a list of all substantial corrections made to *Computers & Typesetting* since the beginning of 2014. (More precisely, it lists errors corrected since the 19th printing of Volume A, the 9th printing of Volume B, the 8th printing of Volume C, the 6th printing of Volume D, and the 7th printing of Volume E. But it omits changes that are "purely cosmetic.") Corrections made to the softcover version of *The TEXbook*, beginning with its 32nd printing, are the same as corrections to Volume A. Corrections to the softcover version of *The METAFONTbook*, beginning with its 11th printing, are the same as corrections to Volume C. Changes to the mini-indexes and master indexes of Volumes B, D, and E are not shown here unless they are not obviously derivable from what has been shown. Some (or all) of these errors have been corrected in the most recent printings.

Page A34, line 3 from the bottom (01/09/20)

---

not, you can say '`I\errorcontextlines=100 \oops`' and try again. (That will usually

Page A43, line 6 (07/24/14)

---

keyboard, or that have been preëmpted for formatting?

Page A49, cummings quote (08/03/19)

(delete the period at the end of the line)

Page A66, line 3 from the bottom (08/26/17)

Such displays of box contents will be discussed further in Chapters 12 and 27.

Page A105, lines 9–16 (01/16/21)

If you say `\vadjust{`⟨ vertical mode material ⟩`}` within a paragraph, TEX will use internal vertical mode to insert the specified material into the vertical list that encloses the paragraph, immediately after whatever line contained the position of the `\vadjust`. For example, you can say '`\vadjust{\kern1pt}`' to increase the amount of space between lines of a paragraph if those lines would otherwise come out too close together. (The author did that in the current line, just to illustrate what happens.) Also, if you want to make sure that a page break will occur immediately after a certain line, you can say '`\vadjust{\eject}`' anywhere in that line.

Page A122, lines 3–8 (11/24/19)

`\count255`, `\dimen255`, `\skip255`, `\muskip255`, and `\toks255` are traditionally kept available for such purposes. Furthermore, plain TEX reserves `\dimen0` to `\dimen9`, `\skip0` to `\skip9`, `\muskip0` to `\muskip9`, and `\box0` to `\box9` for "scratchwork"; these registers are never allocated by the `\new...` operations. We have seen that `\count0` through `\count9` are special, and `\box255` also turns out to be special; so those registers should be avoided unless you know what you are doing.

Page A155, line 8 from the bottom (01/17/21)

`\mathopen{\hbox{$\left#1`⟨ strut ⟩`\right.$}}`

Page A155, the bottom six lines (12/10/18)

dividual symbols; `\left`...`\right` constructions are treated as "inner" subformulas, which means that they will be surrounded by additional space in certain circumstances. All other subformulas are generally treated as ordinary symbols, whether they are formed by `\overline` or `\hbox` or `\vcenter` or by simply being enclosed in braces. Thus, `\mathord` isn't really a necessary part of the TEX language; instead of typing '`$1\mathord,234$`' you can get the same effect from '`$1{,}234$`'.

Page A158, line 19 (12/10/18)

Inner is an inner atom produced by '`\left`...`\right`';

Page A170, lines 18 and 19 (12/10/18)

subformulas delimited by `\left` and `\right` are treated as type Inner. The following table is used to determine the spacing between pairs of adjacent atoms:

Page A171, line 19 from the bottom (06/15/19)

formula produces a result essentially equivalent to '`\left(`⟨ subformula ⟩`\right)`', when

Page A215, line 16 from the bottom becomes two lines (10/13/20)

- Just after a token such as $\$_3$ that begins math mode, to see if another token of category 3 follows.

Page A222, lines 21–23 (01/16/21)

`\hbox`⟨ box specification ⟩`{`⟨ horizontal mode material ⟩`}` (see Chapter 12)
`\vbox`⟨ box specification ⟩`{`⟨ vertical mode material ⟩`}` (see Chapter 12)
`\vtop`⟨ box specification ⟩`{`⟨ vertical mode material ⟩`}` (see Chapter 12)

Page A222, lines 11–13 from the bottom (01/16/21)

ter 15. The `\vsplit` operation is also explained in Chapter 15. In math modes an additional type of box is available: `\vcenter`⟨ box specification ⟩`{`⟨ vertical mode material ⟩`}` (see Chapter 17).

Page A232, line 14 (01/10/21)

tabs outside; '`\global\settabs`' will not do what you might think it should.

Page A233, lines 3–5 (04/27/15)

Only two tabs are set in this case, because only two `&`'s appear in the sample line. (A sample line usually ends with `&\cr`, as it does here, because text material between the last tab and `\cr` isn't used for anything.)

Page A252, lines 5–7 (12/25/20)

blank, and the footline is normally a centered page number, but you can specify any headline and footline that you want by changing the token lists `\headline` and `\footline`. For example,

Page A253, lines 7–9 from the bottom (10/27/20)

`\everypar` or `\errhelp`, except that TEX retains the begin-group symbol '`{`' at the beginning and the end-group symbol '`}`' at the end. These grouping characters help to keep the output routine from interfering with what TEX was doing

Page A256, line 19 (08/28/15)

```
\baselineskip=24pt \lineskiplimit=0pt
```

Page A277, lines 9 and 10 from the bottom (08/26/17)

⟨ hyphenation assignment ⟩ ⟶ `\hyphenation`⟨ filler ⟩`{`⟨ hyphenations ⟩`}`
| `\patterns`⟨ filler ⟩`{`⟨ patterns ⟩`}`

Page A286, bottom two lines (and affecting the top lines of page 287) (08/26/17)

stands for zero or more ⟨assignment⟩ commands other than `\setbox`, possibly with ⟨filler⟩. If the assignments are not followed by a ⟨character⟩, where ⟨character⟩ stands

Page A287, lines 11–17 (04/22/20)

■ `\discretionary`⟨disc text⟩⟨disc text⟩⟨disc text⟩. A ⟨disc text⟩ has the form '⟨filler⟩`{`⟨horizontal mode material⟩`}`', where the material is processed in restricted horizontal mode and should contain only fixed-width things. More precisely, the horizontal list formed by each ⟨disc text⟩ must consist only of characters, ligatures, kerns, boxes, and rules; there should be no glue or penalty items, etc. This command appends a discretionary item to the current list; see Chapter 14 for the meaning of a discretionary item. The space factor is not changed.

Page A292, lines 8–10 (04/22/20)

■ `\discretionary`⟨disc text⟩⟨disc text⟩⟨disc text⟩. This command has the same effect as in horizontal mode (see Chapter 25), but the third ⟨disc text⟩ must produce an empty list.

Page A299, line 11 from the bottom (11/01/20)

is corrupted or was prepared for a different version of TEX.

Page A305, bottom line (06/30/20)

```
\setbox0=\hbox{#1}\advance\dimen0 by -\wd0 }.
```

Page A309, line 2 becomes two lines (12/06/20)

represent text entered from the user's terminal, or with '`<insert>`', when they represent text inserted during error recovery).

Page A316, lines 17 and 18 from the bottom (09/03/15)

(The next line must also not be too tall.) Here `\specialstar` is a box of height zero and depth `\strutdepth`, and it puts an asterisk in the left margin:

Page A320, lines 5–9 from the bottom (06/27/15)

17.21. Assigning `\delcode‘{` would not work to allow '`\left{`', because the brace has category 1 and isn't a legal ⟨delim⟩. Allowing brace delimiters would be a bad idea because it would mess up other constructions, such as arguments to macros, and components of alignments. Moreover, a user who gets away with '`\left{`' is likely to try also '`\bigl{`', which fails miserably.

Page A326, line 12 (08/26/17)

its natural width. The `\hbox` version also invokes `\everyhbox` and `\everymath`.

Page A329, line 3 of answer 20.7 (05/15/19)

the three tokens `!1`, `#2`, `[`$_1$; the ⟨replacement text⟩ consists of the six tokens `{`$_1$, `#`$_6$,

Page A329, line 6 of answer 20.7 (05/15/19)

is otherwise irrelevant. Thus, '`\def\!!1#2#[{##]!!#2]`' would produce an essentially

Page A329, line 5 from the bottom of answer 20.7 (05/15/19)

```
!1<-x
```

Page A329, bottom line of answer 20.7 (05/15/19)

final parameter in the parameter text; '`!1`' would have been rendered '`#1`'.

Page A332, lines 13 and 14 (08/26/17)

21.10. If you say '`{\let\the=0\edef\next{\write\cont{`⟨token list⟩`}}\next}`', the `\write` will be exercuted after `\edef` expands everything except `\the`.

Page A332, bottom line (11/15/19)

```
\+&{\bf end};\cr  % note that the semicolon isn't bold
```

Page A342, lines 12 and 13 (08/14/20)

of plain TEX format; but some of them are primitive (built in), such as '`\par`' (end of paragraph), '`\noindent`' (beginning of non-indented paragraph), and '`/`' (italic

Page A345, lines 10–13 from the bottom (06/27/15)

Braces are used for grouping, when supplying arguments to macros; so they cannot also be used as math delimiters, or as arguments to macros such as `\big`. (One could change their catcodes to 12, and use some other pair of characters for grouping; but that would not be plain TEX.)

Page A346, lines 10–22 (11/24/19)

number identification.) (2) The registers `\count255`, `\dimen255`, `\skip255`, `\toks255`, and `\muskip255` are freely available in the same way. (3) All assignments to the scratch registers whose numbers are 1, 3, 5, 7, and 9 should be `\global`; all assignments to the other scratch registers (0, 2, 4, 6, 8, 255) should be non-`\global`. (This prevents the phenomenon of "save stack buildup" discussed in Chapter 27.) (4) Furthermore, it's possible to use any register in a group, if you ensure that TEX's grouping mechanism will restore the register when you're done with the group, and if you are certain that other macros will not make global assignments to that register when you need it. (5) But when a register is used by several macros, or over long spans of time, it should be allocated by `\newcount`, `\newdimen`, `\newbox`, etc. (6) Similar remarks apply to input/output streams used by `\read` and `\write`, to math families used by `\fam`, to sets of hyphenation rules used by `\language`, and to insertions (which require `\box`, `\count`, `\dimen`, and `\skip` registers all having the same number).

Page A347, line 6 (06/30/20)

```
\def\wlog{\immediate\write-1 } % this will write on log file (only)
```

Page A347, line 10 (11/24/19)

```
\outer\def\newmuskip{\alloc@3\muskip\muskipdef\@cclv}
```

Page A347, line 14 (11/24/19)

```
\outer\def\newtoks{\alloc@5\toks\toksdef\@cclv}
```

Page A350, lines 15 and 16 from the bottom (01/17/21)

format; it shouldn't cost much for people to acquire all the fonts of plain TEX in addition to the ones that they really want. Second, it is desirable on many computer systems to

Page A364, line 5 from the bottom (01/14/21)

```
\def\fmtversion{3.1415926535} % identifies the current format
```

Page A370, lines 11 and 12 (08/26/17)

close as possible to the ASCII conventions. (b) Make sure that codes *´041–´046*, *´060–´071*, *´136*, *´141–´146*, and *´160–´171* are present and that each unrepresentable in-

Page A373, lines 21 and 22 (01/17/21)

and `\if...\fi` tests, as well as special operations like `\the` and `\input`, while the latter category includes the primitive commands listed in Chapters 24–26. The expansion of

Page A375, bottom three lines (06/30/20)

`$$\generaldisplay$$` to be invoked, with `\eq` defined to be α. Furthermore, when an equation number β is present, it should be stored in `\eqn`, and the test `\ifeqno` should be true. In such cases `\ifleqno` should distinguish `\leqno` from `\eqno`. Here

Page A398, lines 4 and 5 (08/26/17)

```
\setbox2=\lastbox \setbox\footins=\vbox{\box2}
```

since `\lastbox` will be the result of `\rigidbalance`, which is an hbox.

Page A407, line 5 from the bottom (06/30/20)

```
\interlinepenalty5000\def\par{\endgraf\penalty5000 }}
```

Page A413, line 11 from the bottom (05/14/19)

The computer file `texbook.tex` that generated *The TEXbook* begins with a

Page A418, line 4 (05/14/19)

TEX commands that look like this in the file `texbook.tex`:

Page A420, line 11 (06/30/20)

```
\def\bull{\vrule height.9ex width.8ex depth-.1ex \relax} % square bullet
```

Page A423, line 16 (06/30/20)

```
  \vrule height6pt depth2pt width0pt \relax} % a strut for \insert\margin
```

Page A445, lines 10–14 (12/10/18)

15e. Enclose the vbox that was constructed in Rule 15c or 15d by delimiters (λ, ρ) whose height plus depth is at least σ_{20}, if $C > T$, and at least σ_{21} otherwise. Shift the delimiters up or down so that they are vertically centered with respect to the axis. Replace the generalized fraction by an Ord atom whose nucleus is the resulting sequence of three boxes (λ, vbox, ρ). Go to rule 19.

Page A446, the bottom three lines of Rule 19 become four lines (01/10/21)

atom and the right boundary item to a Close atom. The entire resulting list now becomes the nucleus of an Inner atom. (All of the calculations in this step are done with C equal to the starting style of the math list; style items in the middle of the list do not affect the style of the right boundary item.)

Page A454, lines 17 and 18 from the bottom (04/13/20)

of the process; the trial word consists of all the letters found in admissible items, up to a maximum of 63. Notice that all of these letters are in font f.

Page A458 and following, selected amendments to the index (01/18/21)

Page Bv (formerly Bvii), bottom two lines (01/15/21)

all of those changes. I now believe that the final bug was discovered on 22 October 2020 and removed in version 3.141592653. The finder's fee has converged to $327.68.

Page B2, line 10 from the bottom (01/15/21)

define *banner* ≡ `´This␣is␣TeX,␣Version␣3.141592653´` { printed when TEX starts }

Page B4, line 8 of §7 (04/02/17)

diagnostic information for `\tracingparagraphs`, `\tracingpages`, and `\tracingrestores`.

Page B21, lines 33 and 34 (04/02/17)

[*´41* → *´46*, *´60* → *´71*, *´136*, *´141* → *´146*, *´160* → *´171*] must be printable. Thus, at least 80 printable characters are needed.

Page B28, lines 3 and 4 (04/02/17)

not serious since we assume that this part of the program is system dependent.

Page B28, line 2 from the bottom (04/02/17)

var k: 0 .. 23; { index to current digit; we assume that $|n| < 10^{23}$ }

Page B35, line 2 of §83 becomes two lines (06/27/20)

loop begin *continue*: **if** *interaction* ≠ *error_stop_mode* **then return**;
clear_for_error_prompt; *prompt_input*("?␣");

Page B36, line 11 of §84 (07/03/20)

"E": **if** *base_ptr* > 0 **then if** *input_stack*[*base_ptr*].*name_field* ≥ 256 **then**

Page B36, line 5 of §85 becomes two lines (07/03/20)

if *base_ptr* > 0 **then**
if *input_stack*[*base_ptr*].*name_field* ≥ 256 **then** *print*("E␣to␣edit␣your␣file."

Page B40, line 5 from the bottom (08/07/20)

("Try␣to␣insert␣an␣instruction␣for␣me␣(e.g.,␣'I\showlists'),")

Page B58, lines 2 and 3 of §136 (10/11/20)

the values corresponding to '\hbox{}'. The *sub_type* field is set to *min_quarterword*, for historic reasons that are no longer relevant.

Page B88, line 16 (10/22/20)

The mode is temporarily set to zero while processing \write texts.

Page B102, lines 3 and following of §241 (12/11/20)

information, something special is needed. The program here simply assumes that suitable values appear in the global variables *sys_time*, *sys_day*, *sys_month*, and *sys_year* (which are initialized to noon on 4 July 1776, in case the implementor is careless).

procedure *fix_date_and_time*;
begin *sys_time* ← 12 * 60; *sys_day* ← 4; *sys_month* ← 7; *sys_year* ← 1776; { self-evident truths }
time ← *sys_time*; { minutes since midnight }
day ← *sys_day*; { day of the month }
month ← *sys_month*; { month of the year }
year ← *sys_year*; { Anno Domini }
end;

Page B103, replacement for §246 (12/11/20)

246. Of course we had better declare a few more global variables, if the previous routines are going to work.

⟨ Global variables 13 ⟩+ ≡
old_setting: 0 .. *max_selector*;
sys_time, *sys_day*, *sys_month*, *sys_year*: *integer*; { date and time supplied by external system }

Page B122, lines 9 and 10 of §291 (10/12/20)

The enclosing `{` and `}` characters of a macro definition are omitted, but an output routine will be enclosed in braces.

Page B143, lines 2, 3, 4 become four lines (01/15/17)

routines that should be aborted, but we can sketch the ideas here: For a runaway definition or a runaway balanced text, we will insert a right brace; for a runaway preamble, we will insert a special `\cr` token and a right brace; and for a runaway argument, we will set *long_state* to *outer_call* and insert `\par`.

Page B188, line 8 (04/02/17)

function *str_toks*(*b* : *pool_pointer*): *pointer*; { converts *str_pool*[*b* .. *pool_ptr* − 1] to a token list }

Page B192, line 17 (10/22/20)

label *found*, *continue*, *done*, *done1*, *done2*;

Page B192, line 3 of §474 (10/22/20)

begin *continue*: *get_token*; { set *cur_cmd*, *cur_chr*, *cur_tok* }

Page B193, line 4 of §476 (05/20/20)

if *cur_tok* < *left_brace_limit* **then**

Page B193, line 10 of §476 becomes two lines (10/22/20)

help2(`"I'm␣going␣to␣ignore␣the␣#␣sign␣you␣just␣used,"`)
(`"as␣well␣as␣the␣token␣that␣followed␣it."`); *error*; **goto** *continue*;

Page B196, line 5 from the bottom (02/17/18)

help1(`"This␣\read␣has␣unbalanced␣braces."`); *align_state* ← 1000000; *limit* ← 0; *error*;

Page B199, lines 1–3 of §494 (10/25/20)

494. Here is a procedure that ignores text until coming to an `\or`, `\else`, or `\fi` at the current level of `\if`...`\fi` nesting. After it has acted, *cur_chr* will indicate the token that was found, but *cur_tok* will not be set (because this makes the procedure run faster).

Page B214, lines 2–6 of §536 (12/11/20)

begin *wlog*(*banner*); *slow_print*(*format_ident*); *print*(`"␣␣"`); *print_int*(*sys_day*); *print_char*(`"␣"`);
months ← `'JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC'`;
for *k* ← 3 ∗ *sys_month* − 2 **to** 3 ∗ *sys_month* **do** *wlog*(*months*[*k*]);
print_char(`"␣"`); *print_int*(*sys_year*); *print_char*(`"␣"`); *print_two*(*sys_time* **div** 60); *print_char*(`":"`);
print_two(*sys_time* **mod** 60);

Page B214, line 2 of §537 becomes two lines (10/29/20)

command is being processed. Beware: For historic reasons, this code foolishly conserves a tiny bit of string pool space; but that can confuse the interactive '`E`' option.

Page B214, bottom line (10/29/20)

if *name* = *str_ptr* − 1 **then** { conserve string pool space (but see note above) }

Page B219, lines 18–20 of §545 (09/19/19)

so-called boundary character of this font; the value of *next_char* need not lie between *bc* and *ec*. If the very last instruction of the *lig_kern* array has *skip_byte* = 255, there is a special ligature/kerning program for a boundary character at the left, beginning at location 256 ∗ *op_byte*+

Page B282, line 1 (and change lines 20–23 accordingly) (04/02/17)

682. Each portion of a formula is classified as Ord, Op, Bin, Rel, Open, Close, Punct, or Inner, for

Page B299, line 4 from the bottom of §722 (10/06/20)

begin *char_warning*(*cur_f*, *qo*(*cur_c*)); *math_type*(*a*) ← *empty*; *cur_i* ← *null_character*;

Page B318, lines 16 and 17 of §761 become one (03/25/19)

fraction_noad: *s* ← *fraction_noad_size*;

Page B333, line 5 of §793 becomes two lines (01/10/20)

cur_loop ← *link*(*cur_loop*); *link*(*p*) ← *new_glue*(*glue_ptr*(*cur_loop*));
subtype(*link*(*p*)) ← *tab_skip_code* + 1;

Page B348, insert a new line after line 5 of §826 (01/15/17)

stat if *tracing_paragraphs* > 0 **then** *end_diagnostic*(*true*); **tats**

Page B348, insert a new line to be the seventh line after the previous change (01/15/17)

stat if *tracing_paragraphs* > 0 **then** *begin_diagnostic*; **tats**

Page B377, line 6 (10/31/20)

hn: 0 . . 64; { the number of positions occupied in *hc*; not always a *small_number* }

Page B417, mini-index (04/02/17)

The entry '*height*, §981.' here and on many later odd-numbered pages should be '*height* = macro, §135.'

Page B522, line 3 of §1306. (10/25/20)

to be in the range $a \le x \le b$. System error messages should be suppressed when undumping.

Page B533, lines 5–8 of §1333. (10/15/20)

loop. (Actually there's one way to get error messages, via *prepare_mag*; but that can't cause infinite recursion.)

If *final_cleanup* is bypassed, this program doesn't bother to close the input files that may still be open.

Page B533, line 12 of §1333. (11/29/20)

begin ⟨ Finish the extensions 1378 ⟩; *new_line_char* ← −1;

Page B534, line 6 of §1335. (11/29/20)

begin $c \leftarrow$ *cur_chr*; **if** $c \neq 1$ **then** *new_line_char* ← −1;

Page B537, line 18 of §1338 becomes two lines (10/05/20)

begin *clear_terminal*;
loop

Page B537, lines 11 and 12 from the bottom of §1338 become three lines (04/02/17)

begin goto *breakpoint*;
{ go to every declared label at least once }
breakpoint: *m* ← 0; `@{'BREAKPOINT'@}`

Page B600, the bottom five lines (05/14/19)

they occupy in a typical production system (executable code size for dark blocks, global data size for light blocks). In this way the chart indicates a total of about $12 \times 22 = 264$K bytes of memory, plus $12 \times 10 = 120$K for the dynamic memory region not shown explicitly. The dynamic memory is often considerably larger in practice, because it is desirable to accommodate large macro packages and large pages.

Page Cx, line 4 from the bottom (06/14/20)

Page C39, lines 10 and 11 become three lines (07/04/20)

that has already been designed. All you'll see is '`(io.mf The letter 0 [79])`' or possibly only '`(io.mf [79])`', followed by '`*`'. Now the fun starts: You should type

Page C68, lines 9, 28, 35, 36, 38 (11/11/17)

```
uniformdeviate -100             -36.1628
z slanted 1/6                   (0.16667y+x,y)
(a,b)zscaled(3,4)               (-4b+3a,3b+4a)
(a,b)zscaled dir 30             (-0.5b+0.86603a,0.86603b+0.5a)
(a,b)dotprod(3,4)               4b+3a
```

Page C72, lines 4–18 (07/16/20)

⟨numeric atom⟩ ⟶ ⟨numeric variable⟩
| ⟨numeric token primary⟩
| `(` ⟨numeric expression⟩ `)`
| `normaldeviate`
| `length` ⟨string primary⟩
| `length` ⟨path primary⟩
| `length` ⟨pair primary⟩
| `angle` ⟨pair primary⟩
| `xpart` ⟨pair primary⟩
| `ypart` ⟨pair primary⟩
| ⟨numeric operator⟩⟨numeric primary⟩
⟨numeric token primary⟩ ⟶ ⟨numeric token⟩ `/` ⟨numeric token⟩
| ⟨numeric token not followed by '`/` ⟨numeric token⟩'⟩
⟨numeric primary⟩ ⟶ ⟨numeric atom not followed by `[` ⟨expression⟩ `,`⟩
| ⟨numeric atom⟩ `[` ⟨numeric expression⟩ `,` ⟨numeric expression⟩ `]`

Page C76, lines 8–16 from the bottom (11/11/17)

tom edge of the type. (With plain METAFONT's **beginchar** each character has a "bounding box" that runs from $(0, h)$ at the upper left and (w, h) at the upper right to $(0, -d)$ and $(w, -d)$ at the lower left and lower right; variable d represents the depth of the type. The values of w, h, and d might change from character to character, since the individual pieces of type need not have the same size in a computer-produced font.)

Page C80, line 14 (06/13/20)

penpos⟨suffix⟩(⟨unknown⟩, ⟨known⟩).

Page C83, line 16 (06/13/20)

```
### 0.5a=-c-0.5b+1.5
```

Page C83, line 19 (06/13/20)

the only dependent variable is now d, which equals $0.5c + 0.75b + 0.75$. (This is

Page C96, line 13 from the bottom (10/31/20)

illustrates the use of $u\#$, $s\#$, $ht\#$, *logo_pen*, *leftstemloc*, o, *xgap*, and *barheight*:

Page C106, lines 19–21 (07/03/20)

pixels. (Some typesetting systems use both of these device-dependent amounts to alter their current position on a page, just after typesetting each character. Other systems, like typical `dvi` software associated with TEX, assume that *chardy* = 0 but use *chardx*

Page C113, lines 5–11 from the bottom (07/20/20)

$s^{\#} := 5pt^{\#}$; **define_pixels**(s); % side of the square
$z_1 = (0,0)$; $z_2 = (s,0)$; $z_3 = (0,s)$; $z_4 = (s,s)$;
for $k = 1$ **upto** 4: $z[k+4] = z[k] + (\frac{2}{3}s, \frac{1}{3}s)$; **endfor**
pickup pencircle scaled $.4pt$; **draw** z_5 -- z_6 -- z_8 -- z_7 -- cycle;
pickup pencircle scaled $1.6pt$; **erase draw** z_2 -- z_4 -- z_3;
pickup pencircle scaled $.4pt$; **draw** z_1 -- z_2 -- z_4 -- z_3 -- cycle;
for $k = 1$ **upto** 4: **draw** $z[k]$ -- $z[k+4]$; **endfor**.

Page C114, line 7 (07/20/20)

for $k = 0$ **upto** 4: $z[k] = \mathit{center} + (\mathit{radius}, 0)$ rotated$(90 + \frac{360}{5}k)$; **endfor**

Page C128, lines 13 and 14 (06/13/20)

changed. Plain METAFONT has a **tensepath** operation that does this. For example, **tensepath** *unitsquare* = $(0,0)$ --- $(1,0)$ --- $(1,1)$ --- $(0,1)$ --- cycle.

Page C136, lines 18 and 19 (07/17/20)

only about 0.28 with respect to the initial and final directions; since METAFONT insists that tensions be at least 0.75, this anomalous path could never have arisen if the control

Page C155, line 7 (10/07/20)

⟨program⟩ ⟶ ⟨statement list⟩⟨statement⟩ `end`

Page C160, lines 7–9 (06/25/20)

might produce a transcript that includes the following diagnostic information:

```
rotatedaround(EXPR0)(EXPR1)->
 shifted-(EXPR0)rotated(EXPR1)shifted(EXPR0)
```

Page C165, lines 5–7 from the bottom (11/11/17)

(i.e., parameters in parentheses), then we name zero or one or two undelimited parameters. Then comes an '=' sign, followed by the replacement text, and **enddef**. The '=' sign might also be ':='; both mean the same thing.

Page C171, lines 18–20 (08/16/20)

Chapter 14's syntax rules for ⟨path primary⟩, via ⟨pair primary⟩. A pair expression is not considered to be of type **path** unless the path interpretation is the only possibility.

Page C176, line 7 from the bottom (07/09/20)

if `@#`($x_{\_}$): $tx_{\_}$ **else**: $fx_{\_}$ **fi** := $x_{\_}$; **endfor**

Page C180, line 3 from the bottom (06/24/20)

'=' or ':=' following **let**.

Page C187, line11 from the bottom (07/12/20)

| `substring` ⟨pair expression⟩ `of` ⟨string primary⟩

Page C189, line 14 (06/13/20)

'`!` ' and followed by '`.`', followed by lines of context as in METAFONT's normal error

Page C200, line 12 from the bottom (08/27/20)

$y_1 = y_2 = good.y(.5[-d, h] + 1.1pt)$;

Page C202, line 17 from the bottom (06/13/20)

command, and it works only when the *penpos* angle is 0. If the *penpos* command is

Page C210, bottom eight lines, and top ten lines of page C211 (07/16/20)

⟨numeric atom⟩ ⟶ ⟨numeric variable⟩ | ⟨numeric argument⟩
| ⟨numeric token primary⟩
| ⟨internal quantity⟩
| `normaldeviate`
| `(` ⟨numeric expression⟩ `)`
| `begingroup` ⟨statement list⟩⟨numeric expression⟩ `endgroup`
| `length` ⟨numeric primary⟩ | `length` ⟨pair primary⟩
| `length` ⟨path primary⟩ | `length` ⟨string primary⟩
| `ASCII` ⟨string primary⟩ | `oct` ⟨string primary⟩ | `hex` ⟨string primary⟩
| ⟨pair part⟩⟨pair primary⟩ | ⟨transform part⟩⟨transform primary⟩
| `angle` ⟨pair primary⟩
| `turningnumber` ⟨path primary⟩ | `totalweight` ⟨picture primary⟩
| ⟨numeric operator⟩⟨numeric primary⟩
| `directiontime` ⟨pair expression⟩ `of` ⟨path primary⟩

⟨numeric token primary⟩ ⟶ ⟨numeric token⟩ `/` ⟨numeric token⟩
| ⟨numeric token not followed by '`/` ⟨numeric token⟩' ⟩

⟨numeric primary⟩ ⟶ ⟨numeric atom not followed by `[` ⟨expression⟩ `,` ⟩
| ⟨numeric atom⟩ `[` ⟨numeric expression⟩ `,` ⟨numeric expression⟩ `]`

Page C214, line 6 becomes two lines (07/17/20)

⟨future pen primary⟩ ⟶ ⟨future pen argument⟩
| `pencircle`

Page C214, line 6 from the bottom (07/12/20)

| **substring** ⟨pair expression⟩ **of** ⟨string primary⟩

Page C217, lines 20–25 (10/07/20)

⟨program⟩ ⟶ ⟨statement list⟩⟨non-title statement⟩ **end**
| ⟨statement list⟩⟨non-title statement⟩ **dump**
⟨statement list⟩ ⟶ ⟨empty⟩ | ⟨statement⟩ **;** ⟨statement list⟩
⟨statement⟩ ⟶ ⟨empty⟩ | ⟨title⟩
| ⟨equation⟩ | ⟨assignment⟩ | ⟨declaration⟩
| ⟨definition⟩ | ⟨compound⟩ | ⟨command⟩

Page C219, line 25 (05/25/20)

to see which of its subscripts and suffixes have occurred. For example, if you're

Page C224, lines 7–9 from the bottom (12/21/18)

```
y4r=-0.9848thinn+259.00049
x4r=-0.08682thinn+144
y4=-0.4924thinn+259.00049
```

Page C226, lines 9 and 10 (11/01/20)

This means that the preloaded base you have specified cannot be used, because it is corrupted or was prepared for a different version of METAFONT.

Page C228, line 27 (06/19/20)

```
l.94 endfor
```

Page C228, line 4 from the bottom (07/12/20)

might want to review now.) You probably also have a **proof** mode diagram:

Page C234, line 4 of answer 4.6 (07/20/20)

for $k = 1$ **upto** 6: $z[k]' = .2[z[k], z_0]$; **endfor**

Page C241, line 2 (11/11/17)

```
\mode=cheapo; input cheaplogo10
```

Page C242, line 11 of answer 13.7 (07/20/20)

for $k = 1$ **upto** 4: $z[k+4] = z[k] + (\frac{2}{3}s, \frac{1}{3}s)$; **endfor**

Page C243, lines 7 and 8 (11/08/15)

draw subpath$(k, k+1)$ of *star*; **cullit**;
undraw subpath$(k+2, k+3)$ of *star* **withpen** *eraser*; **cullit**;

Page C243, line 3 of answer 13.11 (06/17/20)

def overdraw expr c = **begingroup save** *region*;

Page C243, lines 12–16 of answer 13.11 (05/24/20)

beginchar(`"M"`, $1.25in^{\#}, .5in^{\#}, 0$); **pickup pencircle** scaled $.4pt$;
$z_1 = (20, -13)$; $z_2 = (30, -6)$; $z_3 = (20, 1)$; $z_4 = (4, -7)$;
$z_5 = (-12, -13)$; $z_6 = (-24, -4)$; $z_7 = (-15, 6)$;
path M; M = (*origin* .. z_1 .. z_2 .. z_3 .. z_4 .. z_5 .. z_6 .. z_7 ..
origin .. $-z_7$ .. $-z_6$ .. $-z_5$ .. $-z_4$ .. $-z_3$ .. $-z_2$ .. $-z_1$ .. cycle)

Page C246, line 2 of answer 14.13 (08/16/20)

path z_0 `--` z_1 is equivalent to 'z_0 .. controls $1/3[z_0, z_1]$ and $2/3[z_0, z_1]$ .. z_1', and the

Page C247, line 1 of answer 15.5 (06/13/20)

15.5. **beginchar**$(126, 25u^{\#}, h\_height^{\#} + border^{\#}, 0)$; `"Dangerous left bend"`;

Page C247, replacement for answer 15.7 (07/21/20)

15.7. Replace lines 10 and 11 by

pickup pencircle scaled $3/4pt$ yscaled $1/3$ rotated -60;
draw $(z_1 \ldots p)$ transformed t;
addto *currentpicture* **also** *currentpicture*
rotatedaround$((.5w, .5h)$ yscaled *aspect_ratio*$, -180)$;

Page C249, line 1 of answer 18.9 (08/02/20)

18.9. **beginchar**(`"H"`, $13u^{\#}$, "ht"$^{\#}, 0$); **pickup** *broad_pen*;

Page C249, line 11 of answer 18.9 (08/02/20)

filldraw $bot\_serif\_edge_4$

Page C250, line 4 of answer 19.1 (04/19/20)

because it saves a wee bit of time and because ';' often belongs before **endfor**.

Page C250, replacement for answer 19.3 (07/12/20)

19.3. Yes, if and only if $n - \frac{1}{2}$ is an even integer. (Because ambiguous values are rounded upwards.)

Page C251, replacement for answer 22.1 (07/12/20)

22.1 (a) If and only if n is an integer between 0 and 255. (b) If and only if s is a string of length 1.

Page C254, lines 10–13 from the bottom become five lines (06/26/20)

```
? H
I found no right delimiter to match a left one. So I've
put one in, behind the scenes; this may fix the problem.

?
```

Page C260, the "line" after line 3 (06/14/20)

{ `font_size` / `font_slant` / `font_normal_space` / `font_normal_stretch` / `font_normal_shrink` / `font_x_height` / `font_quad` / `font_extra_space` } { `=` / `:=` / ⟨empty⟩ } ⟨numeric#⟩; { `ligtable`⟨ligs/kerns⟩ / `charlist`⟨codes⟩ / `extensible`⟨codes⟩ / `fontdimen`⟨info⟩ / `headerbyte`⟨info⟩ };

Page C261, lines 16 and 17 from the bottom (06/14/20)

{ `proofrule` / `screenrule` } `(`⟨pair⟩`,`⟨pair⟩`);` `makegrid(`⟨numerics⟩`)(`⟨numerics⟩`);`
`proofrulethickness` ⟨numeric#⟩; `proofoffset` ⟨pair⟩.

Page C266, lines 19 and 20 (07/04/20)

You can say either '`incr x`' or '`incr (x)`', within an expression; but neither of them are valid statements by themselves.

Page C269, line 11 (01/10/21)

`\smode="specmode"; mag=`⟨magnification⟩`; input` ⟨font file name⟩

Page C277, lines 15–19 (03/06/17)

```
def openit = openwindow currentwindow from origen    % and please correct
 to (screen_rows,screen_cols) at (-50,300) enddef;   % "(-50,300)" too
def showit_ = display currentpicture inwindow currentwindow enddef;
def showit = openit; let showit=showit_; showit enddef; % first time only
```

Plain METAFONT has several other terse commands similar to '**openit**' and '**showit**':

Page C279, line 1 (11/11/17)

```
 blacker:=.1;                  % make pens a teeny bit blacker
```

Page C289, line 20 (10/07/20)

```
if {{(pair x) cand x>(0,0)}}: A else: B fi.
```

Page C291, line 18 (07/24/20)

```
save u_; setu_ u; let switch_ = if; if false: enddef.
```

Page C292, line 10 from the bottom (10/23/20)

be known by saying '**if** known $(p-q)$: $p = q$ **else**: **false fi**'; transforms could be handled

Page C293, lines 13 and 14 from the bottom (10/27/20)

$f(-1)$ is false! When $c \to 0$, the quantity $a^3 + b^3$ approaches $-\infty$ when c is positive, $+\infty$ when c is negative. An attempt to '*solve* $f(1, -1)$' will divide by zero and come

Page C295, line 2 (07/04/20)

'interpolate $(1, 1)$ .. $(3, 2)$ .. $(15, 4)$ of 7' the approximate value 3.37.

Page C299, bottom four lines of code become five (08/06/20)

```
primarydef t Bernshtein nn = begingroup save r; r =
 begingroup for n=nn downto 2:
  for k=1 upto n-1: u_[[[k]]]:=t[[[u_[[[k]]],u_[[[k+1]]] ]]];
  endfor endfor u_[[[1]]] endgroup; numeric u_[[[]]];
 r endgroup enddef;
```

Page C299, line 5 after the code becomes two lines (08/06/20)

brackets are nested inside of brackets. However, the auxiliary variables '`u_[[[`k`]]]`' must not remain independent at the end.

Page C305, lines 14–18 (07/08/20)

```
width_adj#:=0pt#;          % width adjustment for certain characters
serif_fit#:=0pt#;          % extra sidebar near lowercase serifs
    ⋮
low_asterisk:=false;       % should the asterisk be centered at the axis?
math_fitting:=false;       % should math-mode spacing be used?
```

Page C317, line 21 becomes two lines (11/11/17)

⟨label⟩ ⟶ ⟨code label⟩ | ⟨code⟩ `::` | `:`
⟨code label⟩ ⟶ ⟨code⟩ `:`

Page C318, lines 10–16 from the bottom (11/11/17)

⟨code label⟩⟨labeled code⟩
| ⟨code label⟩⟨labeled code⟩
⟨extensible command⟩ ⟶ **extensible** ⟨code label⟩⟨four codes⟩
⟨four codes⟩ ⟶ ⟨code⟩ **,** ⟨code⟩ **,** ⟨code⟩ **,** ⟨code⟩

Notice that a ⟨code label⟩ can appear in a **ligtable**, **charlist**, or **extensible** command. These appearances are mutually exclusive: No code may be used more than once as a label. Thus, for example, a character with a ligature/kerning program cannot also be **extensible**, nor can it be in a **charlist** (except as the final item).

Page C333, line 29 (10/25/19)

```
"if charcode>0:currentpicture:=currentpicture scaled mg;fi;"
```

Page C333, bottom two lines become one (11/11/17)

```
if unknown scale: scale := max(1,round(pixels_per_inch/300)); fi
```

Page C339, line 3 (05/21/20)

ing ‘ß’, ‘æ’, ‘œ’, and ‘ø’) and the uppercase letters (including ‘Æ’, ‘Œ’, and ‘Ø’) are

Page C341, line 14 from the bottom (11/11/17)

prints the `\table` and the `\text`; `\bigtest` gives you the works, plus a mysterious word

Page C345 and following, selected amendments to the index (01/20/21)

Page Dv, line 16 (01/16/21)

I believe that the final bug in METAFONT was discovered on January

Page Dv, bottom two lines (01/16/21)

corporates all of those changes. I now believe that the final bug was discovered on 03 July 2020 and removed in version 2.71828182. The finder's fee has converged to $327.68.

Page D2, last line of §2 (01/15/21)

define *banner* ≡ `'This␣is␣METAFONT,␣Version␣2.71828182'` { printed when METAFONT starts }

Page D14, line 1 of §30 (05/05/14)

30. The *input_ln* function brings the next line of input from the specified file into available

Page D21, line 8 of §47 (10/11/20)

g: *str_number*; { the string just created }

Page D27, lines 3 and 4 of §61 (04/02/17)

is not serious since we assume that this part of the program is system dependent.

Page D28, line 7 (04/02/17)

var k: 0 .. 23; { index to current digit; we assume that $|n| < 10^{23}$ }

Page D32, line 2 of §78 becomes two lines (06/27/20)

loop begin *continue*: **if** *interaction* ≠ *error_stop_mode* **then return**;
clear_for_error_prompt; *prompt_input*("?␣");

Page D32, line 11 of §79 (07/03/20)

"E": **if** *file_ptr* > 0 **then if** *input_stack*[*file_ptr*].*name_field* ≥ 256 **then**

Page D33, line 5 of §80 (07/03/20)

if *file_ptr* > 0 **then**
if *input_stack*[*file_ptr*].*name_field* ≥ 256 **then** *print*("E␣to␣edit␣your␣file."

Page D37, line 9 of §93 (08/07/20)

("Try␣to␣insert␣an␣instruction␣for␣me␣(e.g.,␣'I␣show␣x;'),")

Page D82, line 2 from the bottom (09/19/19)

define *boundary_char* = 41 { the boundary character for ligatures }

Page D85, lines 3 and 4 of §194 (and §194 actually moves to page D86) (12/11/20)

information, something special is needed. The program here simply assumes that suitable values appear in the global variables *sys_time*, *sys_day*, *sys_month*, and *sys_year* (which are initialized to noon on 4 July 1776, in case the implementor is careless).

Page D85, the final six lines of §194 (and §194 actually moves to page D86) (12/11/20)

procedure *fix_date_and_time*;
begin *sys_time* ← 12 ∗ 60; *sys_day* ← 4; *sys_month* ← 7; *sys_year* ← 1776; { self-evident truths }
internal[*time*] ← *sys_time* ∗ *unity*; { minutes since midnight }
internal[*day*] ← *sys_day* ∗ *unity*; { day of the month }
internal[*month*] ← *sys_month* ∗ *unity*; { month of the year }
internal[*year*] ← *sys_year* ∗ *unity*; { Anno Domini }
end;

Page D86, replacement for §196 (12/11/20)

196. Of course we had better declare a few more global variables, if the previous routines are going to work.

⟨ Global variables 13 ⟩+ ≡
old_setting: 0 .. *max_selector*;
sys_time, *sys_day*, *sys_month*, *sys_year*: *integer*; { date and time supplied by external system }

Page D97, line 2 of §221 (05/26/17)

the definition of attribute nodes) that it is convenient to let *info*(*p*) = 0 stand for '[]'.

Page D148, line 7 (06/12/18)

but the $\log n$ factor is buried in our implicit restriction on the maximum raster size.) The

Page D237, line 5 of §513 (05/26/17)

for $n \leftarrow 0$ **to** $n1 - n0 - 1$ **do** $env\_move[n] \leftarrow mm0$;

Page D250, line 2 of §534 (05/26/17)

direction $\big(right\_u(p), left\_v(q)\big)$; and there's a line of length $\geq delta$ from vertex q to vertex r,

Page D296, line 11 (06/23/20)

name points to the *eqtb* address of the macro being expanded, if the current token list

Page D324, line 13 of §713 (12/20/20)

help2(`"After␣`‘`exitif␣<boolean␣expr>`’`␣I␣expect␣to␣see␣a␣semicolon."`)

Page D326, line 5 from the bottom (06/23/20)

{ invokes a user-defined sequence of commands }

Page D334, lines 1 and 2 of §742 (10/25/20)

742. Here is a procedure that ignores text until coming to an **elseif**, **else**, or **fi** at the current level of **if**...**fi** nesting. After it has acted, *cur_mod* will indicate the token that was found.

Page D339, line 4 of §757 (06/16/20)

(A user who tries some shenanigan like '**for** ... **let endfor**' will be foiled by the *get_symbol*

Page D351, lines 2–7 of §790 become five lines (12/11/20)

begin *wlog*(*banner*); *slow_print*(*format_ident*); *print*(`"␣␣"`); *print_int*(*sys_day*); *print_char*(`"␣"`);
months ← `’JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC’`;
for $k \leftarrow 3 * sys\_month - 2$ **to** $3 * sys\_month$ **do** *wlog*(*months*[*k*]);
print_char(`"␣"`); *print_int*(*sys_year*); *print_char*(`"␣"`); *print_two*(*sys_time* **div** 60); *print_char*(`":"`);
print_two(*sys_time* **mod** 60);

Page D352, line 2 of §793 becomes two lines (10/29/20)

command is being processed. Beware: For historic reasons, this code foolishly conserves a tiny bit of string pool space; but that can confuse the interactive '`E`' option.

Page D352, line 5 from the bottom (10/29/20)

if *name* = *str_ptr* − 1 **then** { conserve string pool space (but see note above) }

Page D354, line 2 from the bottom (07/29/20)

cur_type = *path_type* means that *cur_exp* points to the first node of a path; nobody else points

Page D469, lines 18–20 of §1093 (09/19/19)

so-called boundary character of this font; the value of *next_char* need not lie between *bc* and *ec*. If the very last instruction of the *lig_kern* array has *skip_byte* = 255, there is a special ligature/kerning program for a boundary character at the left, beginning at location 256 ∗ *op_byte*+

Page D469, line 30 of §1093 (01/15/21)

tional halt; no ligature or kerning command is performed.

Page D471, lines 20 and 21 (08/07/20)

param: **array** [1 .. *max_font_dimen*] **of** *scaled*; { **fontdimen** parameters }
np: 0 .. *max_font_dimen*; { the largest **fontdimen** parameter specified so far }

Page D474, line 2 from the bottom (08/07/20)

help1 (`"A␣colon␣should␣follow␣a␣headerbyte␣or␣fontdimen␣location."`); *back_error*;

Page D508, line 3 of §1189. (10/05/20)

to be in the range $a \le x \le b$. System error messages should be suppressed when undumping.

Page D516, line 6 (10/15/20)

If *final_cleanup* is bypassed, this program doesn't bother to close the input files that may still be open.

Page D519, line 17 (01/15/21)

fix_date_and_time; *init_randoms*(*sys_time* + *sys_day* ∗ *unity*);

Page D520, line 18 of §1212 becomes two lines (10/05/20)

begin *clear_terminal*;
loop

Page D520, lines 11 and 12 from the bottom of §1212 become three lines (04/02/17)

begin goto *breakpoint*;
{ go to every declared label at least once }
breakpoint: *m* ← 0; `@{'BREAKPOINT'@}`

Page D566, the bottom five lines (05/14/19)

they occupy in a typical production system (executable code size for dark blocks, global data size for light blocks). In this way the chart indicates a total of about $8 \times 22 = 176$K bytes of memory, plus $8 \times 15 = 120$K for the dynamic memory region not shown explicitly. The dynamic memory is often considerably larger in practice, because it is desirable to accommodate large macro packages and large pictures.