SOFTWARE—PRACTICE AND EXPERIENCE, VOL. 19(7), 607–685 (JULY 1989)

The Errors of TEX*

DONALD E. KNUTH
Computer Science Department, Stanford University, Stanford, California 94305, U.S.A.

SUMMARY

This paper is a case study of program evolution. The author kept track of all changes made to TEX during a period of ten years, including the changes made when the original program was first debugged in 1978. The log book of these errors, numbering more than 850 items, appears as an appendix to this paper. The errors have been classified into fifteen categories for purposes of analysis, and some of the noteworthy bugs are discussed in detail. The history of the TEX project can teach valuable lessons about the preparation of highly portable software and the maintenance of programs that aspire to high standards of reliability.

KEY WORDS Errors Debugging TEX Program evolution Language design True confessions

INTRODUCTION

I make mistakes. I always have, and I probably always will. But I like to think that I learn something, every time I go astray. In fact, one of my favourite poems consists of the following lines by Piet Hein:[1]

> The road to wisdom? Well, it's plain
> and simple to express:
> Err
> and err
> and err again
> but less
> and less
> and less.

I am writing this paper on 5 May 1987, exactly ten years since I began to work intensively on software systems for typesetting. I have certainly learned a lot during those ten years, judging from the number of mistakes I made; and I would like to share what I have learned with other people who are developing software. The best way to do this, as far as I know, is to present a list of all the errors that were corrected in TEX while it was being developed, and to attempt to analyse those errors.

* TEX is a trademark of the American Mathematical Society.

0038–0644/89/070607–79$39.50

Received 19 September 1988
Revised 28 November 1988

When I mentioned my plan for this paper to Paul M. B. Vitányi, he told me about a best-selling book that his grand-uncle had written for civil engineers, devoted entirely to descriptions of foundation work that had proved to be defective. The preface to that book[2] says

> It is natural that engineers should not wish to draw attention to their mistakes, but failures are sometimes due to causes of which there has been no previous experience or of which no information is available. An engineer cannot be blamed for not foreseeing the unknown, and in such cases his reputation would not be harmed if full details of the design and of the phenomena that caused the failure were published for the guidance of others. . . . To be forewarned is to be forearmed.

In my own case I cannot claim that 'unknown' factors lay behind my blunders, since I was totally in control of my programming environment. I can justly be blamed for every mistake I made, and I am certainly not proud of the record. But I see no harm in admitting the horrible truth about my tendency to err, when such details might shed light on the problem of writing large programs. (Besides, I am lucky enough to have a secure job.)

Empirical studies of programming errors, conducted by Endres[3] and by Basili and Perricone,[4] have already led to interesting results and to the conclusion that 'more data must be collected on different projects'. I cannot claim that the data presented below will be as generally applicable as theirs, because all of the programming I shall discuss was done by one person (me). Insightful models of truly large-scale software development and program evolution have been introduced by Belady and Lehman.[5] However, I do have one advantage that the authors of previous studies did not have; namely, the entire program for TEX has been published.[6] Hence I can give fairly precise information about the type and location of each error. The concept of scale cannot easily be communicated by means of numerical data alone; I believe that a detailed list gives important insights that cannot be gained from statistical summaries.

TYPES OF ERROR

Some people undoubtedly think that everything I did on TEX was an error, from start to finish. But I shall consider only a limited class of errors here, based on the log books I kept while I was developing the program. Whenever I made a change, I noted it down for future reference, and it is these changes that I shall discuss in detail. Edited forms of my log books appear in the appendix below.

I guess I could say that this paper is about 'changes', not 'errors', because many of the changes were made in order to introduce new features rather than to correct malfunctions. However, new features are necessary only when a design is deficient (or at least non-optimal). Hence, I will continue to say that each change represents an error, even though I know that no complex system will ever be error-free in this extended sense.

The errors in my log books have each been assigned to one of fifteen general categories for purposes of analysis:

A, an algorithm awry. Here my original method proved to be incorrect or inadequate, so I needed to change the procedure. For example, error no. 212 fixed

a problem in which footnotes appeared on a page backwards: the last footnote came out first.

B, a blunder or botch. Here I knew what I ought to do, but I wrote something else that was syntactically correct—sort of a mental typo. For example, in error no. 126 I wrote 'before' when I meant 'after' and vice versa. I was thinking so much of the Big Picture that I did not have enough brainpower left to get the small details right.

C, a clean-up for consistency or clarity. Here I changed the rules of the language to make things easier to remember and/or more logical. Sometimes this was just a surface change to TEX's 'syntactic sugar', as in error no. 16 where I decided that `\input` would be a better name than `\require`.

D, a data structure debacle. Here I did not properly update the representation of information to preserve the appropriate invariants. For example, in error no. 105 I failed to return nodes to available memory when they were no longer accessible.

E, an efficiency enhancement. Here I changed the program so that it would run faster; the existing code was correct but slow. For example, in error no. 287 I decided to give TEX the ability to preload fount information, since it took a while to read thirty short files at the beginning of every run.

F, a forgotten function. Here I did not remember to do everything I had intended, when I actually got around to writing a particular part of the code. It was a simple error of omission, rather than commission. For example, in error no. 11 and again in no. 172 I had a loop of the form while $p \neq$ null do, and I forgot to advance the pointer p inside the loop! This seems to be one of my favourite mistakes: I often forget the most obvious things.

G, a generalization or growth of ability. Here I realized that some extension of the existing specifications was desirable. For example, error no. 303 generalized my original primitive command '`\ifT` ⟨char⟩' (which tested if a given character was 'T' or not) to the primitive '`\if` ⟨char⟩⟨char⟩' (which tested if two given characters were equal). Eventually, in no. 666, I decided to generalize further and allow '`\if`⟨token⟩⟨token⟩'.

I, an interactive improvement. Here I made TEX respond better to the user's needs. Sometimes I saw how to help TEX identify and recover from errors in the documents it was processing. I also kept searching for better ways to communicate the reasons underlying TEX's behaviour, by making diagnostic information available in symbolic form. For example, error no. 54 introduced '. . .' into the display of context lines so that users could easily tell when information was truncated.

L, a language liability. Here I misused or misunderstood the programming language or system hardware I was working with. For example, in error no. 24 I wanted to reduce a counter modulo 8, so I wrote $t := (t - 1) \bmod 8$; this unfortunately made t negative because of the way mod was defined. Sometimes I forgot the precedence of operators, etc.

M, a mismatch between modules. Here I forgot the conventions I had built into a subroutine when I actually got around to using that subroutine. For example, in error no. 64 I had a macro with four parameters (x_0, y_0, x_1, y_1) that define a rectangle; but when I used it, I gave the parameters in different order, (x_0, x_1, y_0, y_1). Such 'interface errors' included cases when a procedure had unwanted side-effects (such as clobbering a global variable) that I failed to take into

account. Some mismatches (such as incorrect data types) were caught by the compiler and not entered in my log.

P, a promotion of portability. Here I changed the organization or documentation of the program; this affected only a person who would try to read or modify the code, not a person who tried to run it. For example, in error no. 59, one of my comments about how to set the size of memory had '≥' where I meant to say '≤'. (Most changes of this kind were not recorded in my log; I noted only the noteworthy ones.)

Q, a quest for quality. Here I changed the specifications of what the program should output from given input, when I learned how to improve the typographic appearance of the output. For example, error no. 187 changed TEX's behaviour when typesetting formulae that have an unusually complex superscript; as a result, TEX now produces

$$e^{\frac{1}{1-q_j^2}} \qquad \text{instead of} \qquad e^{\frac{1}{1-q_j^2}}.$$

R, a reinforcement of robustness. Whenever I realized that TEX could loop or crash in the presence of certain erroneous input, I tried to make the code bullet-proof. For example, error no. 200 made sure that a user-supplied character number was between 0 and 127; otherwise parts of TEX's memory could be wiped out.

S, a surprising scenario. Errors of type S were particularly bad bugs that forced me to change my original ideas, because of unforeseen interactions between various parts of the program. For example, error no. 25 was logged when I first discovered a consequence of TEX's convention about blank lines denoting the end of a paragraph: There is often a blank space in TEX's internal data structure just before a paragraph ends, because a space is usually supplied at the end of the line just preceding a blank line. Thus I had to write new code to delete the unwanted space. Whenever such unexpected phenomena showed up, I had to go back to the drawing board and fix the design.

T, a trivial typo. Sometimes I did not type the right thing when I entered the program into the computer, although my original pencil draft was correct. For example, in error no. 48 I had typed '−' instead of '+'. If a typing mistake was detected by the compiler as a syntax error, I did not log it, because bad syntax can easily be corrected.

Nine of these categories (A, B, D, F, L, M, R, S, T) represent 'bugs'; such errors absolutely had to be corrected. The other six categories (C, E, G, I, P, Q) represent 'enhancements'; I could have refused to consider the existing situation erroneous. As remarked earlier, I am considering all items in the log to be indications of error. But there is a significant difference between errors of these two kinds: I felt guilty when fixing the bugs, but I felt virtuous when making the enhancements.

My classification of errors into fifteen categories is *ad hoc*, but at the moment it is the best way I can think of to make sense out of my experiences. Some of the bug categories refer to simple flaws in the basic mechanics of programming: writing the right thing but typing it wrong (T); thinking the right thing but writing it wrong (B); knowing the right thing but forgetting to think it (F); imperfectly knowing the tools (L) or the specifications (M). Such bugs are easy to fix once they have been identified. Categories A and D represent the next level of difficulty, as we get into technical

aspects of what programming is all about. (As Niklaus Wirth has said, Algorithms + Data Structures = Programs.) Category R covers the special situation in which we want a program to survive even when its input is incorrect. Finally, category S accounts for higher-level surprises; these are the subtle bugs that result from complex interactions between different parts of a system. Thus the nine types of bugs have a somewhat logical structure. The remaining six categories—cleanliness (C), efficiency (E), generalization (G), interaction (I), portability (P) and quality (Q)—seem to provide a reasonable way to classify the various kinds of enhancements that were made to TEX during its development.

My classification scheme relies more on essential functionality than on the external form of the program. Thus it is not easy to use my statistics about the number of errors per category to answer questions such as 'How many bugs were due to improper use of goto statements?' Such questions are interesting to teachers of programming, but I no longer think that they are extremely important. If I had indexed my errors by syntactic categories, I would have found that error nos. 45, 91, 119, 155, 231, 352, 354, 419, 523, 581 and 801 could be ascribed to my use or abuse of goto; also no. 512 could be added to this list, since return and goto are analogous. Thus we can conclude from my experience with TEX that goto statements can indeed be harmful. On the other hand we must balance this fact with the realization that bad gotos account for only 1·4 per cent of my errors; we must identify other culprits if we're going to do away with the other 98·6 per cent. Sure enough, several other errors were caused by lapses in my use of other control structures: A case statement got me in trouble in no. 21; a while confused me in no. 29; if–then–else led me astray in nos. 467, 471, 680 and 843. (See also nos. 796 and 845, where efficiency of control was important.) I conclude that *every* feature of a programming language can be harmful, if it is misused.

Some of the errors noted in my log book were much more devastating than others. In certain cases the changes were far-reaching, affecting dozens of different parts of the program; several days of 'hacking' were necessary before such changes had been made and verified. For example, change no. 110 required major surgery to the program, because my original ideas were incapable of handling aligned tables inside of aligned tables. On the other hand, some of my errors were only venial sins, and some of the changes were merely twiddles; for example, no. 87 simply improved the wording of a diagnostic message. Although the log does not give an explicit weighting to the errors, the 'heavy' errors tend to cancel with the 'light' ones, so we can still get a reasonable insight into the stability of the program if we calculate, say, the number of errors logged per year.

CHRONOLOGY

The development of TEX has taken place over a period of ten years, and the lessons I learned can best be understood when they are put into the context of the other things I was doing during that time. Typography has many facets, hence TEX itself was only one of the projects I decided to work on. The two most significant companion systems were METAFONT* (a system for typeface design) and Computer Modern (a family of typefaces defined in terms of the METAFONT language); these programs had to be

* METAFONT is a trademark of Addison-Wesley Publishing Company, Inc.

debugged just as TEX did, and their debugging logs show a similar development history. I also needed a dozen or so utility routines to support TEX and METAFONT; the most notable of these are TANGLE and WEAVE, which constitute the WEB system of structured documentation.[7,8]

Beginnings

The genesis of TEX probably took place on 1 February 1977, when I first chanced to see the output of a high-resolution typesetting machine. I was told that this fine typography (the galley proofs of a book by Winston,[9] which our faculty was considering for inclusion in an exam syllabus) was produced by entirely digital methods; yet I could see no difference between the digital type and 'real' type. Therefore I realized that a central aspect of printing had been reduced to bit manipulation. As a computer scientist, I could not resist the challenge of improving print quality by manipulating those bits better. Therefore my diary entry for 8 February says that, already at that time, I began discussing the possibility of new typesetting software with people at Stanford's Artificial Intelligence Lab. By 13 February I had changed my plan to spend a forthcoming sabbatical year in South America; instead of travelling to an exotic place and working on Volume 4 of *The Art of Computer Programming*, I had decided to stay at Stanford and work on digital typography.

I mentioned earlier that the design of TEX was begun on 5 May 1977. A week later, I wrote a draft report containing what I thought was a pretty complete design, and I stayed up until 5 a.m. typing it into the computer. The problem of typesetting seemed quite straightforward, so I soon started thinking about founts instead; I spent the next 45 days writing a program that was destined to evolve into METAFONT. By 28 June, I had 25 lower-case letters in various styles that looked reasonably good to me at the time; and three days later I figured out how to handle the 26th letter, which required some new ideas.[10]

I went back to thinking about TEX on 3 July. Several people had made thoughtful comments on my earlier draft, and I prepared a thoroughly revised language definition after two weeks of further study. (This included two days of working with dictionaries in order to develop an algorithm for hyphenation of English.) The resulting document, I thought, was a reasonably complete specification of a language for typesetting, and I left it in the capable hands of two graduate students who were my research assistants that summer (Frank Liang and Michael Plass). Their job was to implement TEX while I flew off for a visit to China. I returned on 25 August and had just one day to meet with them before leaving on another three-week trip. On 14 September I returned and they presented me with a sheet of paper that had been typeset by their proto-TEX program! They had implemented only about 15 per cent of the language, and they had used data structures that were not general enough or efficient enough to support the remaining 85 per cent; but they had chosen their subset wisely, so that a small test program could run from start to finish. Hence it was easy for me to imagine what a complete system would entail.

Now it was time for Liang and Plass to go back to school, and time for my sabbatical year to begin. I started coding the 'final version of TEX' (or so I thought) on 16 September, and immediately I discovered that their summer work represented a truly heroic achievement. Although I had thought that my specification of TEX was quite complete, I encountered loose ends every 15 minutes or so when I was actually faced

with writing the code. I soon realized that if I had been in my students' shoes—having to implement this language when the author was completely unreachable—I would have thrown up my hands in despair; important policy decisions had to be made at every turn.

That was the first big lesson I learned during my work with TEX: *the designer of a new kind of system must participate fully in the implementation*. Even if I had been available for consultation with my students, they would have had to come to me so often with questions that the work would have dragged on forever. I can imagine them having to spend a half hour or so explaining each particular problem to me, and we would have needed literally hundreds of those meetings. Now I knew why other projects I had heard about, in which the language designer had decided not to be the compiler writer, had failed.

By 14 October I had coded all of TEX except for the parts that typeset mathematics, and except for the routines that convert from TEX's internal representation into codes for an output device. At this point I had to leave for three weeks of travel in Europe. This European trip had been planned long before, so it was mostly unrelated to typesetting; but I did have some interesting discussions about curve-drawing with mathematicians I met in Oberwolfach, Germany, and in Oslo, Norway. I also was able to arrange a visit to the headquarters of Monotype Corporation in Redhill, England.

After returning, I spent November finishing the numerals, upper-case letters, and punctuation marks of the first-draft Computer Modern types. I needed to have a complete fount because I had been invited to give a lecture about this work to the American Mathematical Society, and I did not want to have only lower-case examples to show. I prepared the AMS lecture[11] during December and presented it in January, so I did not have a chance to resume the coding of TEX until 14 January. But finally I was able to write the following in my diary on 9 February 1978:

> Finished the TEX programs including all loose ends and got them all compiled without syntax errors (4 a.m.).

TEX was the first fairly large program I had written since 1970; so it was my first non-trivial 'structured program', in the sense that I wrote it while consciously applying the methodology I had learned in the early 1970s from Dijkstra, Hoare, Dahl and others. I found that structured programming greatly increased my confidence in the correctness of the code, while the code still existed only on paper. Therefore I could wait until the whole program was written, before trying to debug any of it. This saved a lot of time, because I did not have to prepare 'dummy' versions of non-existent modules while testing modules that were already written; I could test everything in its final environment. Of course I had a few qualms in January about whether my code from September would really work; but that gave me more of an incentive to finish the whole thing sooner.

Even on 10 February, when TEX had been compiled and was ready to be tested, I did not feel any compelling need to try it immediately. I knew that the program was fairly readable and 'informally proved correct', so I spent the next month making italic, greek, script, symbols and large delimiter founts. My test program for TEX required those founts, so I did not want to start testing until everything was in place. Again, I knew I was saving time by not having to prepare prototypes that would merely simulate the real thing; structured programming gave me the courage to wait until the whole

system was ready. I finished the large symbols on 8 March, and I happily penned the following in my diary on 9 March:

> Entered all accumulated corrections to TEX program and compiled it—tomorrow the debugging begins!

My log book for errors in TEX began that next day, 10 March; the debugging process will be discussed below. By 29 March I had decided that TEX was essentially working,

> . . . (except perhaps for error recovery)—it's time to celebrate!

I began tuning up the founts and drafting ideas for a user manual; then I spent a few days at Alphatype Corporation in Illinois, from whom Stanford had decided to purchase a phototypesetter. From 11 April to 11 May I took time off from typography to work on dozens of updates to *Seminumerical Algorithms*, which is Volume 2 of *The Art of Computer Programming*;[12] I wanted to incorporate new research results into that text, which was to be TEX's first big application. Then on 14 May I began to get TEX running again; proof copies of pages iv to 8 of Volume 2 came out of our Xerox Graphics Printer on 15 May.

My work was cut out for me during the next weeks: I became a production user of TEX, typing the manuscript of Volume 2. This proved to be an invaluable experience, as explained below. By the time my sabbatical year ended, on 24 September, I had finished the typing up to page 441 of that 700-page book. Improvements to TEX kept occurring to me all during that time, of course—except during a month-long vacation trip with my family. (Even on vacation I kept seeing founts everywhere and thinking about how to draw such letterforms by computer. I spent one morning sitting by one of the trails in the Grand Canyon designing the algebraic notation for METAFONT; my founts had previously been written in a primitive macro language and compiled directly into machine code, not interpreted.) I also spent three weeks that summer writing the first manual for TEX.

Although my sabbatical year was over, I kept working on typography in odd moments between classes in the autumn; the text of Volume 2 was completed on the morning of 15 November. On 17 November I began writing METAFONT, and my diary entry for 31 December 1978 was this:

> Finished the METAFONT interpreter, just in time to celebrate New Year's eve (11:59 p.m.).

Other people had begun to use TEX in August of 1978, and I was surprised to see how fast the system was propagating. I spent my spare time during the first three months of 1979 thinking about how to make TEX available in Pascal form. (The original program was written in SAIL, a language that was available on only a few computers.) During this period I began to experiment with the typesetting of Pascal programs; I wrote a program called BLAISE that converted Pascal source code into a TEX file for pretty-printing. BLAISE soon developed into a system called DOC for structured documentation, completed on 31 March, 1979; programs in DOC format could be converted either to Pascal or to TEX. Luis Trabb Pardo and Ignacio Zabala

subsequently used DOC to prepare a highly portable version of TEX in Pascal, completed in April of 1980.

About this time I learned another big lesson: *writing software is much harder than writing books*. I could not simultaneously teach classes well and finish what needed to be done on typography. So I asked to be excused from teaching in the spring of 1979; my diary for March 22 said,

> Now my obligations are fairly well cleared away and it's back to the stalled research on TEX.

(It turned out that I was able to teach during only 13 of the 21 academic quarters between my sabbatical years in that period. I continued to supervise graduate students, but I gave no classroom lectures during 1983 when the work on TEX and METAFONT was at its peak; I also missed three months in 1982, 1984 and 1985. I really enjoy teaching, but I could not see any way to finish the TEX project without relinquishing almost all of my other duties.)

On 1 April 1979, I returned to METAFONT, which had been written but not debugged. METAFONT began to work on 28 April. Then I began to design software for the Alphatype machine; that took about three months. During the summer I wrote the METAFONT manual, which gave me further experience with TEX. And TEX also received an important stimulus from the American Mathematical Society that summer, when several people (including Barbara Beeton and Michael Spivak) were given the opportunity to spend some time at Stanford developing TEX macros. The AMS people introduced me to several important applications, such as the indexes to *Mathematical Reviews*, which stretched TEX to its limits and led to substantial improvements.

Endings

By 14 August 1979, I felt that TEX was essentially complete and fairly stable. I lectured that evening to about 100 participants of the Western Institute for Computer Science in Santa Cruz, telling about my experiences developing and debugging the program. At that time my log book of errors had accumulated 420 items; little did I know that the final total would be more than twice that! But already I knew that I had learned a lot by keeping the log, and I must have been enthusiastic because I lectured from 7:30 to 9:30 p.m. (The audience was equally enthusiastic—they kept asking me questions until 11:30 p.m. So I resolved to write a paper about the errors of TEX, and at last I am able to do so.)

I devoted the last months of 1979 and the first months of 1980 to Computer Modern, which needed to be rewritten in terms of the new METAFONT. Then I needed to update Volume 2 again—computer science marches inexorably forward—until I had finally finished producing camera-ready copy on our Alphatype. This was the goal I had hoped to achieve during my sabbatical year; I reached it at 2 a.m. on 29 July 1980, about two years late. During the rest of 1980 I wrote papers about what I thought were the most novel ideas in TEX[13] and in METAFONT.[14]

But my research on TEX was by no means finished. About 50 people from all over the U.S.A. met at Stanford on 22 February 1980, and established the TEX User Group (TUG). I asked them if they would mind my cleaning up the language in several upward-incompatible ways, even though this would make the user manual and

their existing computer files obsolete; and nobody objected to such changes! Soon TUG grew dramatically, under the able chairmanship of Richard Palais, and it became international. I realized that I could not disappoint all these people by leaving TEX in its current state and returning immediately to work on subsequent volumes of *The Art of Computer Programming*.

I needed to work out a better 'endgame strategy', and it soon became clear what ought to be done: the original versions of TEX and METAFONT should be scrapped, once they had served their purpose of accumulating enough user experience to indicate what such languages ought to be. New versions of TEX and METAFONT should be written, designed to last a long time and to be highly portable between computers and typesetting devices of all kinds. Moreover, these new programs should be published, because TEX was making it possible to improve the state of the art of program documentation. I decided to do my best to produce a stable system and to explain all I knew about it, so that other people could take it over and maintain it if it proved to be important. This way I could return to other pursuits in good conscience, knowing that if my typographic research had any merit it would be carried on by others in whatever ways would prove to be necessary.

So that was my new goal; I thought I could achieve it in one or two more years. The original TEX program was renamed TEX78, and the new one was to be called TEX82.

Classes and miscellaneous chores kept me too busy to do much else during the first half of 1981, but I began to write TEX82 on 22 August. By 9 September I realized that the DOC system needed to be completely revised, so I spent two months replacing it by a much better system called WEB[8]. Since then my programming language of choice has been WEB (which, unlike DOC, was written in its own language). After a month in Europe, I was able to resume writing TEX82 on 1 December 1981. The draft of TEX82 was completed on 29 June 1982; as before, I wrote the entire program before trying to run any of it.

Meanwhile I had other problems to worry about. When my new copy of *Seminumerical Algorithms* arrived in January 1981, I had expected to be filled with joy at the consummation of so much hard work. Instead, I burned with disappointment, as I realized that I still had a great deal to learn about founts. The early Computer Modern typefaces were not at all what I had hoped to achieve, when I first saw them in print. They had looked reasonably good at low resolution, so I had blithely assumed that high resolution would be much better. Not so. My education in typefaces was barely beginning. Later in 1981 I met Richard Southall, a professor of type design who had exactly the expertise I was lacking; so I invited him to visit Stanford. We spent the entire month of April 1982 working about 16 hours a day, revising Computer Modern from A to z.

I debugged TEX82 in the summer of 1982, then began to write the new manual—called *The TEXbook*[15]—in October. The first manual had been written hastily and finished in 21 days, but I wanted *The TEXbook* to meet much higher standards. Therefore I was not able to finish it until a full year later.

It was during this period, October 1982 to October 1983, that TEX became a mature system. I had to rethink every aspect of its design as I rewrote the manual. Fortunately I was aided by a wonderful group of knowledgeable volunteers, who would meet with me for two or three hours every Friday noon and we would discuss the trade-offs of every important decision. The diverse backgrounds of these people provided an important

counterweight to my one-sided views. Finally, on 9 December 1983, I decided that the first phase of my endgame strategy was complete; I gratefully hosted a coming-of-age party for TEX, with 36 guests of honour, at the Fuki-Sushi restaurant in Palo Alto.

The rest is history. I wrote METAFONT in WEB between December 1983 and July 1984; I wrote *The METAFONTbook* between August 1984 and October 1985, taking off five months (February to July) to rewrite Computer Modern in terms of the new METAFONT. I began another sabbatical year in October 1985, just after the TEX project disbanded. Finally, after adding a few more finishing touches, I was able to celebrate the long-planned completion of my 'endgame' on 21 May 1986, when my publishers sponsored a reception at the Computer Museum in Boston; that was the day I first saw the five hardcover volumes of *Computers & Typesetting*, the books that summarize my nine years of work on TEX, METAFONT and Computer Modern.

Another year has gone by and I would like to report that TEX has proved to be 100 per cent correct. But I cannot, not yet. For I stumbled across a hidden TEX anomaly last January. And I have just been teaching a course about software development based on the internal structure of TEX; students in the class have noticed a few things that should be improved. So I suppose there is still at least one bug lurking there. I plan to hold off publishing this paper until another year or so has gone by, so that I will have more reason to believe that my log book of errors is complete.

CONTENTS OF THE LOG BOOKS

As I said, the appendix to this paper reproduces the entire list of errors that I kept as TEX was evolving. The best way to comprehend how TEX evolved is to peruse this list. The first 519 items refer to the original program TEX78, which was written in SAIL, from the time I began to debug it to the time I stopped maintaining it. The remaining items, numbered 520–849 (as of May 1987), refer to the 'real' program TEX82, which was written in WEB. I did not keep any record of errors removed during the hectic period when TEX82 was being debugged, but items 520 and following include every change that was made to TEX82 after it passed its first test. The differences between TEX78 and TEX82, seen from a user's standpoint, have been listed elsewhere.[16]

I have tried to edit the log entries so that they can be understood in terms of the published listing[6] of TEX82. For example,

15 Add the forgotten case 'set_font:' to eq_destroy. §275 F

is entry no. 15. My original log entry referred to case '`[font]`' in '`eqdestroy`' using SAIL syntax, but I have changed to Pascal syntax in the edited log. Similarly, the 1978 identifier `font` eventually became set_font, so I have adopted the published equivalent. TEX82 contains a procedure called eq_destroy in §275 of the program, and this procedure is quite similar to the `eqdestroy` of TEX78; so I have supplied §275 as a program reference. (It turns out that eq_destroy no longer needs a 'set_font:' subcase, but it did in 1978.) The 'F' after §275 means that this was a bug of type F, a forgotten function.

Changes to a program often spawn other changes later. I have tried to indicate that

phenomenon in the appendix by prefixing the number of a prior error when it was an important part of the reason for a subsequent error. Thus no. 67 is

25 ↦ **67** Replace the space at paragraph end by fillglue, not by zero. §816 B

Error no. 25 was logged when I had been surprised to find a space at the end of TEX's internal representation of a paragraph. I had 'cured' the problem by converting the space from a normal interword space to a space of width zero. But that was not good enough, since it was possible for TEX to try breaking a line at the zero-width space. A better solution was to replace the space by the glue that is always added to fill out the end of a paragraph.

Figure 1 shows a time chart of the first 519 log entries—the errors of TEX78. There is a burst of activity right near the beginning, since I logged the first 237 errors during the three weeks of initial debugging. Thus the main line in Figure 1, which shows the cumulative number of errors as a function of time, is nearly horizontal at the beginning. But it is nearly vertical at the end, since only 13 changes were made during the last year of TEX78's activity.

Another line also appears in Figure 1: it represents the total number of different pages I typeset with TEX78 as I was experimenting with the first version. The dotted line in July 1978 stands for the 200 pages of the first TEX manual, and the dotted line in June 1979 stands for the 100 pages of the first METAFONT manual; the remaining solid lines stand for the 700 pages of Volume 2 and some experiments with DOC.

Figure 1 shows that four different phases can be distinguished in the development of TEX78. First came the debugging phase (Phase 0), already mentioned. Then came a longer period of time (Phase 1) when I typeset several hundred pages of Volume 2 and the first user manual; this experience suggested many amendments to my original design. Then TEX suddenly had more than one user, and different kinds of errors began to show up. *New users find new bugs*. This coming-out phase (Phase 2) included small bursts of changes when I faced new applications—a suite of difficult test cases posed by the American Mathematical Society, then the application to Pascal formatting, then the complex index to *Mathematical Reviews*. Finally there was Phase 3, when changes were made in anticipation of a future TEX82; I wanted several new ideas to be well tested before I programmed the 'ultimate' TEX.

THE INITIAL DEBUGGING STAGE

Let us roll the clock back now and look more closely at the earliest days of TEX78. In some ways this was the most interesting time, because the whole concept of TEX was just beginning to take shape. Figure 2 is a modified version of Figure 1, redrawn with a time warp. There is now exactly one error per time unit, so the 18-day debugging phase has been slowed down to almost half of the total development time; on the other hand, the years 1981–1982 at the bottom go by so fast as to be barely visible.

I mentioned that TEX78 was entirely coded before I first tried to run it on 10 March. My debugging strategy was to walk through the program using the BAIL debugger, a system program by John Reiser that allowed me to execute the statements of my program one at a time; BAIL would also interpret additional SAIL statements that I entered on-line. Whenever I came to a section of program that I had seen before,

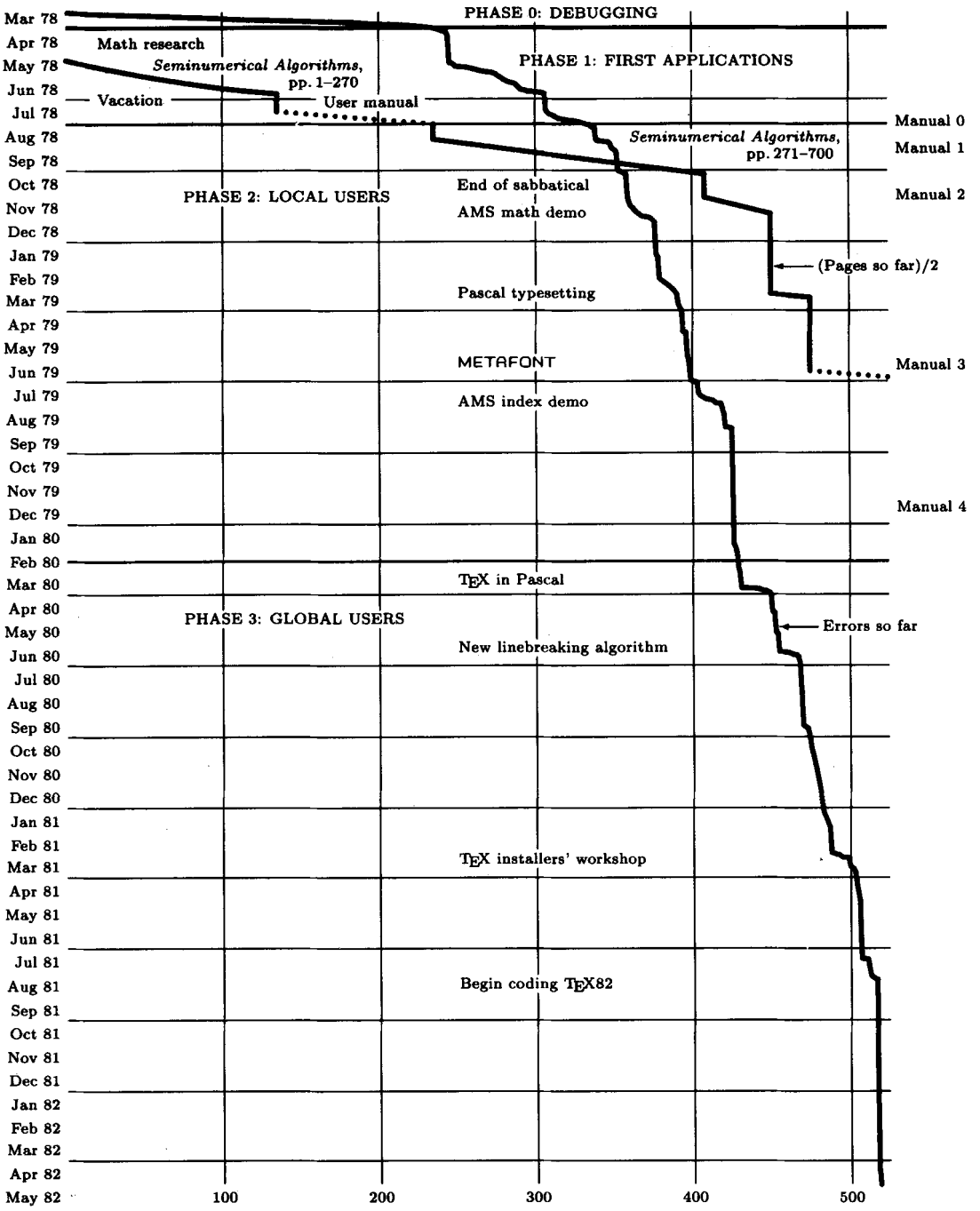

Figure 1. The rise and fall of TEX78

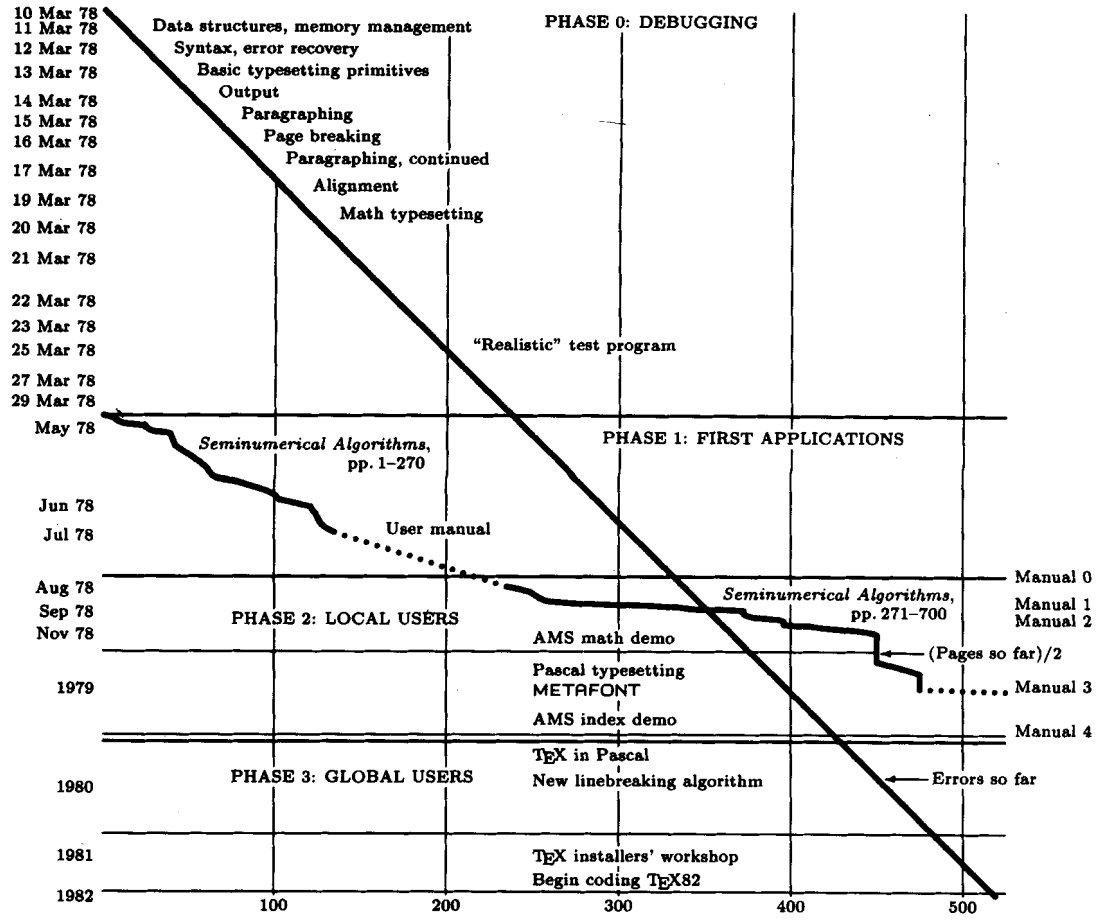

Figure 2. The errors of TEX78

I could set a break-point and continue at high speed until coming to new material. Watching the program execute itself in this 'dynamic order' has always been insightful for me, after I have desk-checked it in the 'static order' of my original code.

Figure 2 shows that I got through the program initialization the first day; then I was gradually able to check out the routines for basic data management, parsing and error reporting. On the fourth day TEX began to combine boxes and glue, and there was visible output on the fifth day. During the following three days I tested the algorithms for breaking paragraphs into lines and breaking lines into pages. All this went rather smoothly; I had already logged 101 errors during this first week, but all of the problems were comparatively minor oversights, to be expected in any program of this size.

On the ninth day I tackled alignment of tables, and got a big shock: my original algorithms were quite wrong. I had greatly misunderstood this aspect of TEX, because I had greatly underestimated the complications of nested alignments. (The log mentions some of the puzzlement and frustration I felt at the time.) I wrestled with alignment for two days before finding a solution.

Then I looked at the last remaining part of TEX, the code for typesetting mathematics; this took another four days. (Well, the 'days' were nights actually; I worked during the night to avoid delays due to time-sharing.) Finally I had seen essentially all of TEX in operation, and I could let it run at full speed instead of relying on single-step mode. I spent six more days helping TEX get through its first test data; finally the test was passed. Whew! The debugging phase was over, 18 days and 237 log-book entries after it began.

I kept track of how long this process took, so that I'd be better able to estimate the duration of future programming projects. Table I gives the figures.

The total debugging time, 132 h, was extremely encouraging to me, because it was much less than the 41 days it had taken me to write the program. Previously I had needed to devote about 70 per cent of program development time to debugging, but now the figure had dropped to about 30 per cent. I considered this to be a tremendous victory for structured programming, since my programming time had also decreased from what it had been with old habits. Later, with the WEB system, I noticed even further gains in productivity.

How big was TEX at the time? I estimated this by counting the number of semicolons (4857) and the number of occurrences of the SAIL reserved words comment (480) and else (223). Since I always put semicolons before end, the total number of statements in the program could be computed as

$$; - \text{comment} + \text{else} = 4857 - 480 + 223 = 4600$$

Thus the debugging strategy I used allowed me to verify about 35 statements per hour.

The fact that I made 237 log entries in 132 h means that I was logging things only about once every 33 min; thus the total time needed to keep the log was negligible. I can definitely recommend the practice to everybody. During most of the debugging time I was clicking away at the keys of my terminal, getting to know exactly what TEX was doing; I needed only a few extra minutes to make the log entries, which helped me to get to know myself.

EARLY TYPESETTING EXPERIENCE

Now that TEX was able to typeset its test program, I could proceed to my main goal, the typesetting of Volume 2. This was a somewhat tedious task—the keyboarding of a

Table I

Day	Time, h	Day	Time, h
10 March 1978	6	19 March 1978	7·5
11 March 1978	7	20 March 1978	10
12 March 1978	8	21 March 1978	8
13 March 1978	7	22 March 1978	6
14 March 1978	8	23 March 1978	7·5
15 March 1978	8	25 March 1978	7
16 March 1978	7	26 March 1978	6
17 March 1978	7	27 March 1978	8
18 March 1978	8	29 March 1978	6

700-page book is not one of life's greatest pleasures—but the regular appearance of nice-looking pages kept me happy. The jagged line in Figure 2 shows my progress in terms of pages typeset versus errors in the TEX log; a similar (even more jagged) line appears in Figure 1, showing pages typeset as a function of time.

The most striking thing about the jagged line in Figure 2 is that it is almost straight. Ideas about how to improve TEX kept occurring to me quite regularly as I typed the manuscript. Between 13 May and 22 June I processed about 250 pages, and added 69 new entries to the log. Those 69 entries included 29 'bugs' and 40 'enhancements'; thus, I thought of a new way to improve TEX at a regular rate of about one enhancement for every six pages typed.

I mentioned earlier my firm conviction that I could not have correctly delegated the coding of TEX to another person; I had to be doing it myself, because writing a new sort of program implies continually revising the specifications. Similarly, I could not have correctly delegated these initial typing experiments to another person. I had to put myself in the rôle of a regular user; there is no substitute for such experience, when a new system is being designed.

But at the time I was not thinking about creating a system that would be used widely; I was designing TEX primarily for my own use. The idea that TEX could or should be generalized to other applications besides *The Art of Computer Programming* dawned on me only gradually, as people kept noticing what I was doing and expressing an interest in it.

John McCarthy observed during this period that TEX was doing a reasonable job with respect to traditional mathematical copy, but he suspected that I would have a tough time typesetting a book about TEX itself. 'That will be the real test', he said, 'because you'll have to shut off many of TEX's automatic features in order to handle problems of self-reference'.

In July I succumbed to John's challenge and prepared a user manual for TEX. Sure enough, this experience helped me identify quite a few weaknesses in the existing design, things that I probably wouldn't have noticed if I had confined my attention to *The Art of Computer Programming* alone. Again I thought of enhancements at the rate of about one for every six or seven pages, as I wrote the manual; but these were not really occasioned by defects in TEX's ability to be self-referential, as John had predicted. The new enhancements came about because the process of manual-writing forced me to think about TEX as a whole, in a new way. The perspective of a teacher/expositor helped me to notice several inconsistencies and shortcomings.

Thus, I came to the conclusion that the designer of a new system must not only be the implementor and the first large-scale user; *the designer should also write the first user manual*. The separation of any of these four components would have hurt TEX significantly. If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important.

PHASES 2 AND 3: USERS

But a system cannot be successful if it is too strongly influenced by a single person. Once the initial design is complete and fairly robust, the real test begins as people with many different viewpoints undertake their own experiments. At the beginning of August, I distributed 45 copies of the draft manual to people who had expressed

interest in using TEX and who had promised to give me feedback before the 'real' user manual would be issued in September. So TEX had a multitude of users for the first time, and I began to learn about a wide variety of new applications and perceptions.

I continued to typeset the remaining 450 pages of Volume 2, and my personal experiences with those pages continued to suggest regular improvements to TEX until I got up to about page 500. But the final 200 pages were just drudgework, not really inspirational to me in any way as far as TEX was concerned. Nor did I learn much more, except about page layout, when I typed the METAFONT manual some months later. The really important influences on TEX after the first manual was published were the users, first because they made different kinds of mistakes than I had anticipated, and later because they had important suggestions about how to improve TEX's capabilities.

Guy Steele was visiting Stanford that summer; he took a copy of TEX back to MIT with him, and I began to get feedback from two coasts. One of Guy's suggestions, which I staunchly resisted at the time, was to include some sort of mini-programming language in TEX so that users could do numerical calculations. Slowly but surely I began to understand the need for such features, which eventually became a basic part of TEX82. Another early user was Terry Winograd, who pushed TEX's early macro capabilities to their limits. He and Michael Spivak, who began to work with TEX in the summer of 1979, taught me a lot about the peculiar properties of macro expansion. Researchers at Xerox PARC also had a significant influence on TEX at this time; Lyle Ramshaw modified the program to work with Xerox's new founts and new output devices, while Leo Guibas and Doug Wyatt undertook to rewrite TEX in the MESA language.

Figures 1 and 2 indicate that the first TEX user manual was issued in five versions. 'Manual 0' was the preliminary draft, handed out to 45 guinea pigs who agreed to help me test the very first system. 'Manual 1' was a Stanford technical report issued a month later; it was reprinted as 'Manual 2' in November, using the higher-resolution printing devices at Xerox PARC. The American Mathematical Society published a paperback version[17] of Manual 2 in the summer of 1979; that was 'Manual 3'. Then Digital Press published 'Manual 4', which included the METAFONT manual and some background information, in December 1979.[18]

The publishers of manuals 3 and 4 asked readers to mail a reply card if they were interested in forming a TEX User's Group, and more than 100 people answered Yes. So the first TUG meeting, in February 1980, marked the beginning of yet another phase in the life of the SAIL program TEX78. A great influx of new users and new applications made me strive for a more complete language. Hence there was a flurry of activity at the end of March 1980, when I decided to extend TEX in more than a dozen ways. These extensions represented only a fraction of the ideas that had been suggested, but they seemed to provide all the requested functions in a clean way. The time was ripe to make the extensions now or never, because the first versions of TEX in Pascal were due to be released in April.

The last significant batch of changes to TEX78 were made in the summer of 1980, when TEX acquired the ability to typeset paragraphs with arbitrary shapes. Still, the error log shows that I kept adding enhancements regularly as the world-wide use of TEX continued to grow. It turned out that the final bugs corrected in TEX78 were all introduced by recent enhancements; they were not present in the program of 1978.

The most significant pattern to be found among the enhancements made to TEX78 after its earliest days is the 'unbundling' of things that used to be frozen inside the

code. At first I had fairly rigid ideas about how much space to put in certain places, about how much penalty to charge for certain line breaks, about how to interpret various characters in the input, and even about where to find certain characters in founts. One by one, starting already at change no. 104, these things became parameters that could be changed by users who had different requirements and/or different preferences.

THE REAL TEX

I had vastly underestimated the complexities and subtleties of typesetting when I had naïvely expected to work out a complete system for myself during a single sabbatical year. By 1980 it became clear that I had acquired almost a moral obligation to advance the art and science of typography in a more substantial way. I realized that I could never be happy with the monster I had created unless I started over and built an entirely new system, using the experience I had gained from TEX78.

I began writing the new system in the summer of 1981, and I decided to call it TEX82 because I knew it would take a year to complete. Once again I could not delegate the job to an associate; I wanted to rethink every detail of TEX, and I wanted to have a thorough taste of 'literate programming' before I dared to inflict such ideas on others.[7] I wanted to produce truly portable software that would have a chance to serve for many years as a reliable component of larger systems. I wanted TEX82 to justify the confidence that people were placing in TEX78, which was getting more praise than it deserved.

Figure 3 shows the development of TEX82, starting at the moment I decided that it was essentially bug-free; this illustration uses the same time-warp strategy as Figure 2.

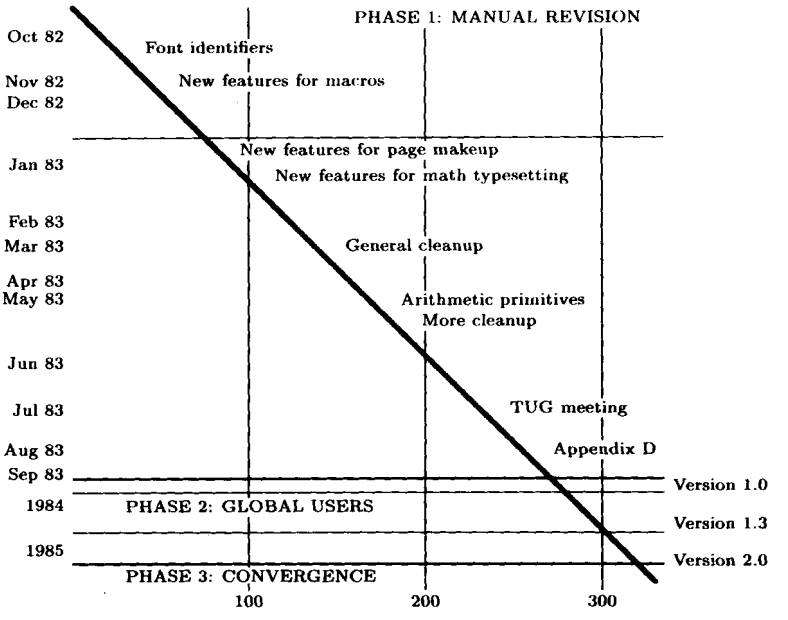

Figure 3. The errors of TEX82

From the beginning there were hundreds of users, so TEX82's Phase 1 was analogous to TEX78's Phase 2. But now there was yet a new dimension: several dozen people were also reading the code and making well-informed comments on how to improve it. Furthermore I had regular meetings with volunteer helpers who represented many different points of view. So I had a golden opportunity to hone the ideas to a new state of perfection.

Two major changes were installed very early in TEX82's history. One was to the way founts are selected in a document (change no. 545), and the other was to the treatment of conditional parts of macros (change no. 564). Both of these changes impinged on many of the fundamental assumptions I had made when writing the code; these were definitely the most traumatic moments in TEX's medical history. I was glad to see that WEB's documentation facilities helped greatly to make such drastic revisions possible.

Phase 1 of TEX82 ended about a year after it began, when I completed writing *The TEXbook*. The log reveals that most of the changes made to TEX during 1983 relate to the chapters of the manual that I was writing at the time. This was the period when TEX really grew up. As I said above, manual writing provides an ideal incentive for system improvements, because you discover and remove glitches that you cannot justify in print. When you are writing a user manual, you also have your last chance to make any enhancements that you have thought about before; if certain enhancements are not made then, you know that you will forever wish you had taken time to add them.

As with TEX78, the error log of enhancements to TEX82 shows a significant trend toward greater user control. More and more things that were originally hardwired in the system became parametric instead of automatic.

Phase 2 of TEX82 began with the paperback publication of *The TEXbook* and ended with the publication of the hardcover edition. During this phase (which lasted from October 1983 to May 1986) I was mostly working on METAFONT and Computer Modern, so TEX changed primarily in ways that would blend better with those systems. The log entries of Phase 2, nos. 790 to 840, also show that a number of ever-more subtle bugs were detected by ever-more sophisticated users during this time. There was also a completely unsubtle bug, no. 808, which somehow had sneaked through all my tests and caused no apparent harm for an amazingly long time.

Now TEX82 is in its third and final phase. It has grown from the original 4600 statements in SAIL to 1376 modules in WEB, representing about 14,000 statements in Pascal. Five volumes describing the complete systems for TEX, METAFONT and Computer Modern have been published. No more changes will be made except to correct any bugs that still might lurk in the code (or perhaps to improve the efficiency or portability, when it is easy to do so while correcting a real bug). I hope TEX82 will remain stable at least until I finish Volume 7 of *The Art of Computer Programming*.

TEST PROGRAMS

Since 1960 I have had extremely good luck with a method of testing that may deserve to be better known: instead of using a normal, large application to test a software system, I generally get best results by writing a test program that no sane user would ever think of writing. My test programs are intended to break the system, to push it to its extreme limits, to pile complication on complication, in ways that the system programmer never consciously anticipated. To prepare such test data, I get into the

meanest, nastiest frame of mind that I can manage, and I write the nastiest code I can think of; then I turn around and embed that in even nastier constructions that are almost obscene. The resulting test program is so crazy that I could not possibly explain to anybody else what it is supposed to do; nobody else would care! But such a program proves to be an admirable way to flush the bugs out of software.

In one of my early experiments, I wrote a small compiler for Burroughs Corporation, using an interpretive language specially devised for the occasion. I rigged the interpreter so that it would count how often each instruction was interpreted; then I tested the new system by compiling a large user application. To my surprise, this big test case did not really test much; it left more than half of the frequency counts sitting at zero! Most of my code could have been completely messed up, yet this application would have worked fine. So I wrote a nasty, artificially contrived program as described above, and of course I detected numerous new bugs while doing so. Still, I discovered that 10 per cent of the code had not been exercised by the new test. I looked at the remaining zeros and said, 'Shucks, my source code wasn't nasty enough, it overlooked some special cases I had forgotten about'. It was easy to add a few more statements, until eventually I had constructed a test routine that invoked all but one of the instructions in the compiler. (And I proved that the remaining instruction would *never* be executed in *any* circumstances, so I took it out.)

I used such 'torture tests' to debug three compilers during the 1960s. In each case very few bugs were ever discovered after the tests had been passed, so the methodology was quite effective. But when I debugged TEX78, my test program was quite tame by comparison—except when I was first testing the mathematics routines (20–23 March). I guess I was not trying as hard as usual to make TEX a bullet-proof system, because I was still thinking of myself as TEX's main user. My original test program for TEX78 was written with an 'I hope it works' attitude, rather than 'I bet I can make it fail'. I suppose I would have found several dozen of the bugs that showed up later (such as nos. 240 and 263) if I had stuck to the torture-test methodology. Still, considering my mood at the time, I suppose it was a good idea to have a test program that would look like real typography; I did not know what TEX should do until I could judge the æsthetic quality of its output.

At any rate, my first test program was based on a sampling of material from Volume 2. I went through that book and boiled it down to five pages that illustrated just about every kind of typographical difficulty to be found in the entire volume. (The output of this test program can be seen in another paper,[19] where David Fuchs and I used the same test data to study some algorithms for fount management.)

Years later, when TEX82 was ready to be debugged, I understood pretty clearly what the program was supposed to do, so I could then apply the superior torture-test methodology. My test program was called TRIP; I spent about five days preparing the first draft of TRIP in July 1982. Here, for example, is a relatively tame part of the original TRIP code:

```
\def\gobble#1{} \floatingpenalty 100
\everypar{A\insert200{\baselineskip400pt\splittopskip\count15pt
  \hbox{\vadjust{\penalty999}}\hbox to -10pt{}}\showthe\pagetotal
  \showthe\pagegoal\advance\count15by1\mark{\the\count15}%
  \splitmaxdepth-1pt\paR\gobble}%abort every paragraph abruptly
```

```
\def\weird#1{\csname\expandafter\gobble\string#1
  \string\csname\endcsname} \message{\the\output\weird\one}
```

(Please do not ask me what it means.) Since then I have probably spent at least 200 hours modifying and maintaining TRIP, but I consider that time well spent, and I think TRIP is one of the most significant products of the TEX project.[20] The reason is that the TRIP test has detected extremely subtle bugs in hundreds of implementations of TEX, bugs that would have been almost impossible to track down in any other way. TEX82, with its TRIP test, has proved to be much more reliable than any of the Pascal compilers it has been compiled with. In fact, I believe it is fair to say that TEX82 has helped to flush out at least one previously unknown compiler bug whenever it has been ported to a new machine or tried on a compiler that has not seen TEX before! These compiler errors were detectable because of the TRIP test. Later I developed a similar test program for METAFONT, called TRAP,[21] and it too has helped to exorcise dozens of compiler bugs.

A single test program cannot detect all possible mistakes. For example, TEX might terminate with a 'fatal error' in several ways, only one of which can happen on any particular run. Furthermore, TRIP runs almost automatically, so it does not test all of TEX's capability for on-line interaction. But TRIP does exercise almost all of TEX's code, and it does so in tricky combinations that tend to fail if any part of TEX is damaged. Therefore it has proved to be a great time-saver: whenever I modify TEX, I simply check that the results of the TRIP test have changed appropriately.

The only difficulty with the TRIP methodology is that I must check the output myself to see if it is correct. Sometimes I need to spend several hours before I have determined the appropriate output; and I am fallible. So TEX might give the wrong answer without my being aware of it. This happened in bug nos. 543 and 722, when I learned to my surprise that TEX had never before done the correct thing with TRIP. A system utility for comparing files suffices now to convince me that incremental changes to TEX or TRIP cause the correct incremental changes to the TRIP test output; but when I began debugging, I needed to verify by hand that thousands of lines of output were accurate.

I should mention that I also believe in the merit of formal and informal correctness proofs. I generally try to prove my programs correct, informally, by stating appropriate invariants in my documentation and checking at my desk that those relations are preserved. But I can make mistakes in proofs and in specifying the conditions for correctness, just as I make mistakes in programming; therefore I do not rely entirely on correctness proofs, nor do I rely entirely on empirical test routines such as TRIP.

LOCATION AND TYPE OF ERRORS

Let me review again the fifteen classes of errors that are listed in my error log:

A — Algorithm	F — Forgotten	P — Portability
B — Blunder	G — Generalization	Q — Quality
C — Cleanup	I — Interaction	R — Robustness
D — Data	L — Language	S — Surprise
E — Efficiency	M — Mismatch	T — Typo

I mentioned before that each of the errors listed in the appendix refers where possible to its approximate location in the program listing of TEX82. It is natural to wonder whether the errors are uniformly interspersed throughout the code, or if certain parts were particularly vulnerable. Figure 4 shows the actual distribution. No part of the program has come through unscathed—or, shall we rather say, unimproved—but some parts have seen significantly more action. The boxes to the left of the vertical lines in Figure 4 represent 'bugs' (A, B, D, F, L, M, R, S, T), whereas the boxes to the right represent 'enhancements' (C, E, G, I, P, Q). The most unstable parts of TEX78 were the parts I understood least when I began to write the code, namely mathematical formatting and alignment. The most unstable parts of TEX82 were the parts that differed most from TEX78 (the conditional instructions and other aspects of macro expansion; also the increased user access to registers and internal quantities used in TEX's decision-making).

I should mention why hyphenation is almost never mentioned in the log of TEX78. Although I said earlier that TEX78 was entirely written before any of it was tested, that is not quite true. The hyphenation algorithm was quite independent of everything else and easily isolated from the code, so I had written and debugged it separately during three days in October 1977. (There is obviously no advantage to testing independent programs simultaneously; that leads only to confusion. But the rest of TEX was highly interdependent, and it could not easily be run when any of the parts

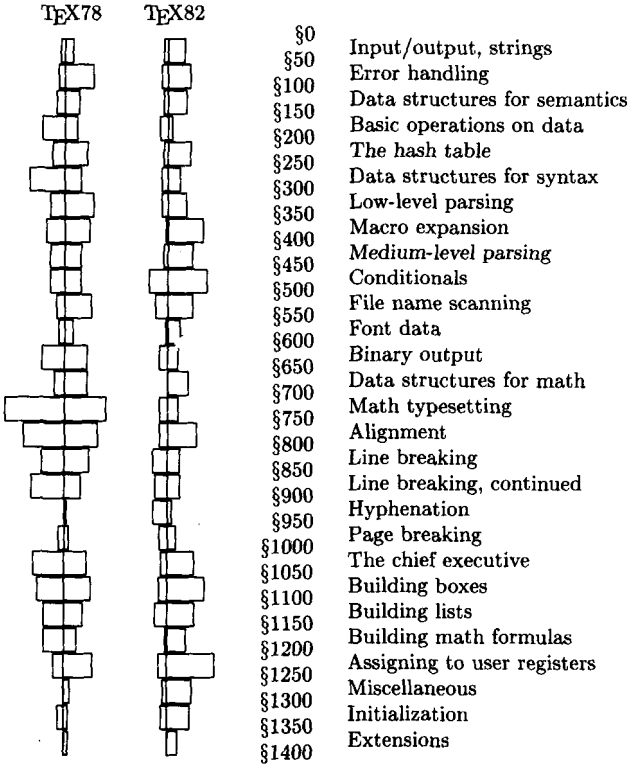

Figure 4. Distribution of [bugs | enhancements] by program location

were absent, except for the routines that produced the final output.) The hyphenation algorithm of TEX78 was English-specific; Frank Liang, who had helped me with this part of TEX78, developed a much better approach in his thesis,[22] and I ultimately incorporated his algorithm in TEX82.

Figure 5 shows the accumulated number of errors of each type in TEX78, with bugs at the bottom and enhancements at the top. Initially the log entries are mostly bugs, with occasional enhancements of type I; at the end, however, enhancements C, G and Q predominate. Figure 6 is a similar diagram for TEX82. In the latter case the vast majority of errors are enhancements, and there are no bugs of types M or T. That is because the debugging phase of TEX82 does not appear in the log, not because I learned how to make fewer mistakes.

SOME NOTEWORTHY BUGS

The gestalt of TEX's evolution can best be perceived by scanning through the log book, item by item. But I would like to single out several errors that were particularly instructive or otherwise memorable.

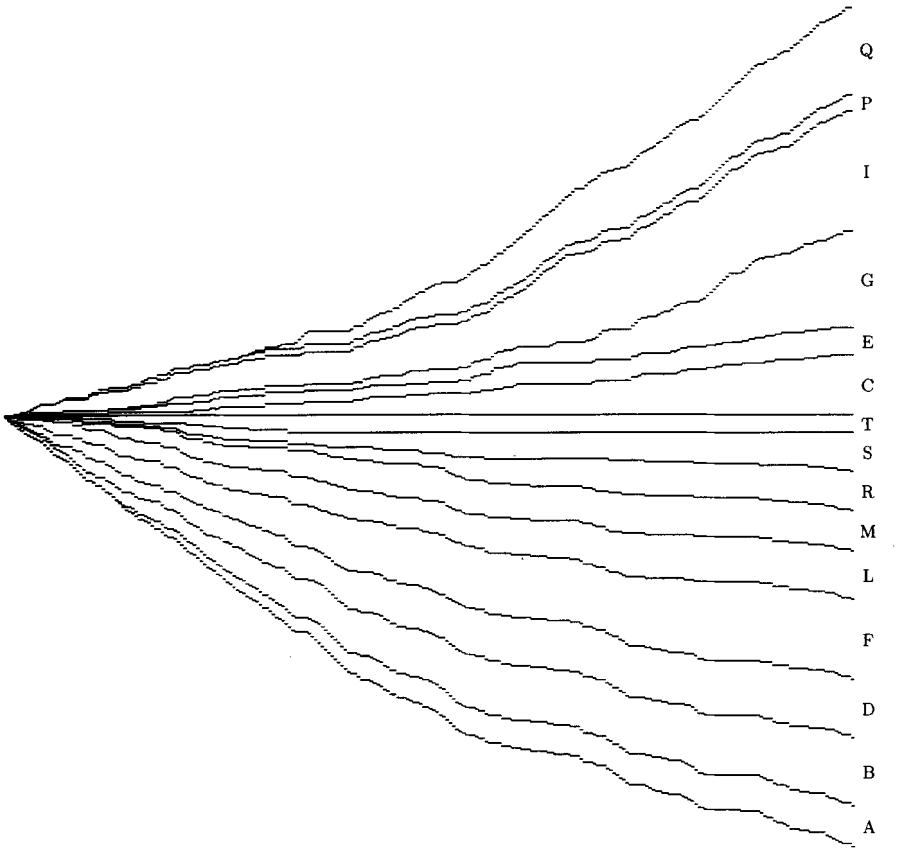

Figure 5. Accumulated errors of TEX78, divided into fifteen categories

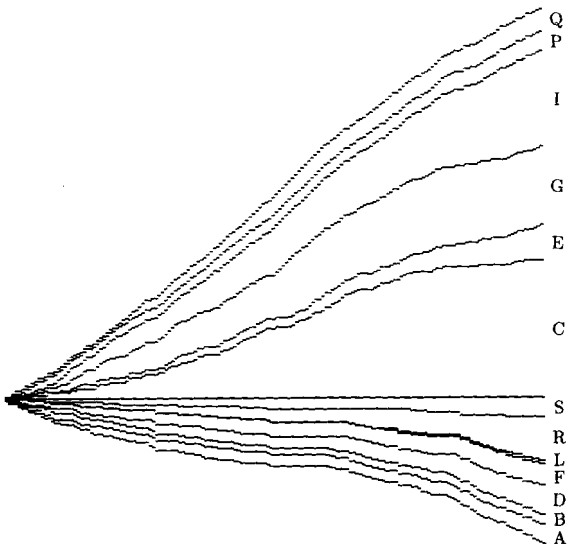

Figure 6. Accumulated errors of TEX82, not counting its initial debugging stage

A, Algorithmic anomalies

I decided from the beginning that the algorithms of TEX would be in the public domain. But if I were to change my mind and charge a fee for my services in inventing them, I would probably request the highest price for a comparatively innocuous-looking group of statements now found in sections 851 and 854 of the program. This precise sequence of logical tests, used to control when a line break is being forced because there is no 'feasible' alternative, has the essential form

```
if α1 ∨ α2 then
  if α3 ∧ α4 ∧ α5 ∧ α6 then σ1
  else if α7 then σ2 else σ3
else σ4
```

and most of the appropriate boolean conditions α_i were discovered only with great difficulty. The program now warns any readers who seek to improve TEX to 'think thrice before daring to make any changes here'. Some indications of my struggles with this particular logic appear in error nos. 75, 93 and 506.

TEX's line-breaking algorithm determines the optimum sequence of breaks for each paragraph, in the sense that the total 'demerits' are minimized over all feasible sequences of breaks. The original algorithm was fairly simple, but it continued to evolve as I fiddled with the formula used to calculate demerits. Demerits are based on the 'badness' b of the line (which measures how loose or tight the spacing is) and the 'penalty' p for the break (which may be at a hyphen or within a mathematical formula). A penalty might be negative to indicate a good break. The original formula for demerits in TEX78 was

$$D = \max(b + p, 0)^2$$

error no. 76 replaced this by

$$D = \begin{cases} (1 + b + p)^2, & \text{if } p \geq 0 \\ (1 + b)^2 - p^2, & \text{if } p < 0 \end{cases}$$

The extra constant 1 was used to encourage paragraphs with fewer lines; the subtraction of p^2 when $p < 0$ gave fewer demerits to good breaks. This improved formula was published on page 1128 of the article on line-breaking by Knuth and Plass.[13] The first draft of TEX82 added an obvious generalization to the improved formula by introducing a `\linepenalty` parameter, l, to replace the constant 1. A further improvement was made in change no. 554, when I realized that better results would be obtained by computing demerits as follows:

$$D = \begin{cases} (l + b)^2 + p^2, & \text{if } p \geq 0 \\ (l + b)^2 - p^2, & \text{if } p < 0 \end{cases}$$

Otherwise, a line with, say, $(b, p) = (50,100)$, followed by a line with $(b, p) = (0,0)$, would be considered inferior to a pair of lines with $(b, p) = (0,100)$ and $(100,0)$, although the second pair of lines would actually look much worse.

B, Blunders

A typical blunder, among the 50 or so errors of class B in the appendix, is illustrated by error nos. 7 and 92. I had declared two symbolic constants in my program, new_line (for one of the three states of TEX's lexical scanner) and next_line (for the sequence of ASCII codes carriage_return and line_feed, needed in SAIL output conventions). Although the meanings were quite dissimilar, the names were quite similar; therefore I confused them in my mind. The compiler did not detect any syntax error, because both were legal in an output statement, so I had to detect and correct the bugs myself. I could have avoided these errors by using a name like cr_lf instead of next_line; but that sounds too jargony. A better alternative would have been new_line_state instead of new_line.

D, Data disasters

My most striking error in data-structure updating was no. 630, which crept in when I made change no. 625. The error needs a bit of background information before I can explain it: using an idea of Luis Trabb Pardo, I was able to save one bit in each node of TEX's main data structures by putting the nodes in which the bit would be 0—the so-called charnodes—into the upper part of the mem array, all other nodes into the lower part. (It was very important to save this bit, because I needed at least 32 additional bits in every charnode.) One of the aspects of change no. 625 was to optimize my data structure for representing mathematical subformulae that consist of a single letter. I could recognize and simplify such a subformula by looking for a list that consisted of precisely two elements, namely a charnode followed by a 'kern node' (for an 'italic correction'). A kern node is identified by (a) not being a charnode, i.e. not having a high memory address, and (b) having the subfield type = 11.

I forgot to test condition (a). But my program still worked in almost every case, because unsuitable lists of length 2 are rare as subformulae, and because the type subfield of a charnode records a fount number. Amazingly, however, within one week of my installing change no. 625, some user happened to create a mathematical list of length 2 in which the second element was a character from fount number 11!

This example demonstrates that I was lucky to have a wide variety of users. Still, such a bug might survive for years before it would cause trouble for anybody.

F, Forgetfulness

As I am writing this paper, I am trying to remember all the points I wanted to explain about TEX's evolution. Probably I will forget something, as I did when I was writing the program for TEX.

Usually a bug of class F was easily noticed when I first looked at the corresponding part of the code, with my walk-through-in-execution-order method of debugging. But I would like to mention two of the F errors that were among the most difficult to find. Both of them occurred in routines that had worked correctly the first few times they were exercised; indeed, these routines had been called hundreds of times, with perfect results, so I no longer suspected that they could be the source of any trouble.

Error no. 91 occurred in the memory allocation subroutine, the first time I ran out of memory. That subroutine had the general form

```
begin ⟨Get ready to search⟩;
repeat ⟨Look at an available slot⟩;
  if ⟨big enough⟩ then goto found;
  ⟨Move to next slot⟩;
until ⟨back at the beginning⟩;
found: ⟨Allocate and return, unless the available list becomes exhausted⟩;
ovfl: ⟨Give an overflow message⟩;
end
```

The bug is obvious: I forgot to say 'goto ovfl' just before the label 'found:'. And it is also obvious why this bug was hard to find: I had lost my suspicions that this subroutine could fail, but when it did fail it allocated one node right in the middle of another. My linked data structure was therefore destroyed, but its defective fields did not cause trouble until several hundred additional operations had been performed by the parts of the program where I was still looking for bugs.

Error no. 203 was even more difficult to find; it lurked in TEX's get_next routine, the subroutine that is executed far more than any other. Whenever TEX is ready to see another token of input, get_next comes into action. Therefore, by the time I had corrected 200 errors, get_next had probably obtained the correct next token more than 100,000 times; I considered it rock-solid reliable.

Since get_next is part of TEX's 'inner loop', I had wanted it to be efficient. Indeed, I learned later that the very first statement of get_next, 'cur_cs ← 0', is performed more often than any other single statement of TEX82. (Empirical tests covering a period of more than a year show that 'cur_cs ← 0' was performed more than 1·4 billion times on Stanford's SUAI computer. The get_avail routine, which is next in importance, was invoked only about 438 million times.) Knowing that get_next was critical, I had tried

to avoid performing 'cur_cs ← 0' in my first implementations, in cases where I knew that the value of cur_cs would not be examined by the consumers of get_next's tokens. In fact, I knew that cur_cs would be irrelevant in the vast majority of cases. (But I also knew, and forgot, Hoare's dictum that premature optimization is the root of all evil in programming.)

Well, you can almost guess the rest. When I corrected my serious misunderstanding of alignments, error nos. 108 and 110, I introduced a new case in get_next, and that new case filled my thoughts so much that I forgot to worry about the 'cur_cs ← 0' operation. Still, no harm was done unless cur_cs was actually being looked at; TEX would not fail unless \cr occurred in an alignment having a special sort of template that required back-up in the parser. As before, the effect of this error was buried in a data structure, where it remained hidden until much later. I found the bug only by temporarily inserting new code that continually monitored the integrity of the data structures. (Such code later became a standard diagnostic feature of TEX82; it can be seen for example in section 167.)

L, Language lossage

Some of my errors (nos. 98, 295, 296, 480) were due to the fact that algorithms involving floating-point numbers sometimes fail because of round-off errors. (I have assigned these errors to class L instead of class A, although it was a close call.) TEX82 was designed to be portable so that it gives essentially identical results on all computers; therefore I avoided floating-point calculations in critical parts of the new program.

Two other errors in my log belong unambiguously to class L: in nos. 63 and 827, I failed to insert parentheses into a macro definition. As a result, when I used the macro with text replacement, any frequent user of macros can guess what happened. (Namely, in no. 827, I had declared the macro

```
hi_mem_stat_min ≡ mem_top − 13
```

and used it in the statement

```
dyn_used ← mem_top + 1 − hi_mem_stat_min;
```

this gave a minus where I wanted a plus.)

M, Mismatches

When I write a program I tend to forget the exact specifications of its subroutines. One of my frequent flubs is to blur the distinction between an object and a pointer to that object. In TEX82, for example, I noticed when I got to error no. 79 that I had called vpackage (p, . . .) where p pointed to the first node of a vlist, whereas in the declaration of vpackage I had assumed parameters of the form (h, . . .) where h points to a list header; thus link(h), not h itself, was assumed to point to the first list item. The compiler did not catch the error because both h and link(h) were of type pointer.

While fixing this bug it occurred to me that vpackage was an oft-used subroutine and that I might have made the same mistake more than once. So I looked closely at each of the 26 places I had called vpackage, and the results proved that I was remarkably

inconsistent: I had specified a list head 14 times, and a direct pointer 12 times! (Fortunately there was not a 13–13 split; that would have been unlucky.)

This error reminded me that I should always check the entire program whenever I notice a mistake; *failures tend to recur*. In fact, several errors of TEX82 (nos. 803, 813, 815, 837) were first noticed when I was debugging similar portions of METAFONT.

R, Robustness

Most of the changes of type R were introduced to keep TEX from crashing when users supply input that does not obey the rules. But some of the Rs in the log are intended to keep TEX alive even when other parts of TEX are failing, because of my programming errors or because somebody else is trying to produce a new modification of TEX.

Thus, for example, in nos. 99 and 123, I redesigned two of my procedures so that they would produce a symbolic printout of given data structures in memory even when those data structures were malformed. I made it possible to obtain meaningful output from arbitrary bit configurations in memory, so that while debugging TEX I could interactively look at garbage and guess how it might have arisen.

One of the most recent changes to TEX, no. 846, has the same flavour: the parameter to show_node_list was redeclared to be of type integer instead of type pointer, because buggy calls on show_node_list might not supply a valid pointer.

S, Surprises

The most serious errors were those due to my global misunderstandings of how the system fits together. The final error in TEX78 was of type S, and I suppose the final error of TEX82 will be yet another surprise.

Let me mention just two of these. The first is extremely embarrasing, but it makes a good story. TEX produces DVI files as output, where DVI stands for DeVice Independent. The DVI language is like a machine language, consisting of 8-bit instruction codes followed in certain cases by arguments to the instructions. Two of the simplest instructions of DVI language are push (code 141) and pop (code 142). It turns out that TEX might output push followed immediately by pop in various circumstances, and this needlessly clutters up the DVI file; so I decided to optimize things a bit by checking to see whether the final byte in my output buffer was push before TEX would output a pop. If so, I could cancel both instructions. This technique even made it possible to detect and cancel long redundant sequences such as push push pop push push pop pop pop. Naturally, I checked to see that the buffer had not been entirely cancelled out when I tested for such an optimization. (I was not 100 per cent stupid.) But I failed to realize that the byte just preceding pop might just happen to be 141 (the binary code for push) when it was the final operand byte of some *other* instruction. Ouch!

The other S bug I want to discuss is truly an example of global misunderstanding, because it arose in connection with my misperceptions about \global definitions in TEX documents. Users can define control sequences such as \abc inside a TEX 'group', which is essentially a 'block' in the sense of Algol scope rules. At the end of a group, local definitions are rescinded and control sequences revert to the meanings

they had at the beginning of the group. In my first implementation of TEX78 I went even further: If \abc was defined inside a group but not before the group had begun, I actually removed \abc from the hash table when the group ended.

There is one exception, however, to TEX's local scope rules (and it is usually the exceptions that lead to surprises). Users can state that a definition is \global; this means that the new definition will survive at the end of the current group, unless it has been globally redefined again. Therefore my implementation removed control sequences from the hash table at group endings only when they had not been globally defined.

That caused bug no. 422, which was identical to one of the first serious bugs I had ever encountered when learning to program in the 1950s: deletions from an 'open' hash table might make other keys inaccessible, unless the deletions occur in FIFO order, or unless the deletion algorithm takes special precautions to relocate keys in the table. (See my book *Sorting and Searching,*[22] pp. 526–527, where I say—in italics—'*The obvious way to delete records from a scatter table doesn't work.*') Alas, I had deleted the control-sequence records in the 'obvious way' in TEX78, not realizing that global definitions destroyed the FIFO order.

To fix bug no. 422, I could not patch the definition procedure by using Algorithm 6.4R from my book,[22] because the organization of TEX did not allow for relocation of keys. So I needed to change the hash table algorithm from linear probing to chaining, which supports arbitrary deletions. This change was not as painful as it might have been at this late date (August 1979), because I had needed an excuse anyway to overcome my initial hash table design. In order to keep the original implementation simple, I had decided to require that control sequence names be essentially unique when restricted to their first six letters. Such a restriction was quite reasonable when I was to be the only user of TEX; but it was becoming intolerable when the number of users began to grow into the thousands. Therefore change no. 422 not only altered the hash discipline, it also changed the entire representation mechanism so that identifiers of arbitrary length could be accommodated.

And that was not the end of the story. Another year and a half went by before I realized (in no. 493) that TEX allows declarations like

```
\def\abc{...}
\global\def\xyz{...\abc...}
```

within a group. In such cases I could not eliminate \abc from the hash table at the end of the group, because a reference to \abc still survived within \xyz. I finally decided not to delete *anything* from the hash table (although I did provide a mechanism to prevent unwanted keys from ever getting in; see nos. 294 and 769).

How did such serious bugs remain undetected for so long? They lay dormant because normal usage of TEX does not require complicated interactions between local and global definitions in groups. Most formatting is simpler than this; even complex books such as *The Art of Computer Programming* and the TEX manual itself do not need such generality. But if I had used the TRIP test methodology in the early days, I would have found and corrected the local/global problems right at the start. This experience suggests that all software systems be subjected to the meanest, nastiest torture tests imaginable; otherwise they will almost certainly continue to exhibit bugs for years after they have begun to produce satisfactory results in large applications.

T, Typographical trivia

The typographical errors of TEX were not especially significant, but I will mention two of them (nos. 69 and 86), where my original SAIL code looked like this:

```
glueshrink(q)← glueshrink(q)← glueshrink(t);
x← x← width(q).
```

SAIL was written for the extended ASCII character set that once was widely used at Stanford, MIT, CMU and a few other places; one of the important characters was '←', for Algol's ':='. The language allowed multiple assignment, hence both of these statements were syntactically correct (although rather silly).

A language designer straddles a narrow line between restrictiveness and permissiveness. If almost every sequence of characters is syntactically correct, the inevitable typographical errors will almost never be detected. But if almost no sequences of characters are syntactically correct, typing becomes a real pain.

In TEX78 I made a terrible decision (no. 402) to allow users to type a letter such as 'A' whenever TEX was expecting to see a number; the meaning was to use the ASCII code of A (97) as the number. This extended the language for certain hacker-type applications; but it caused all sorts of grief to ordinary users, because their typographical errors were being treated as perfectly meaningful TEX input, and they could not figure out what was going wrong. (I compounded the error in no. 507; see also no. 511. This is a sorry part of the record.) TEX82 resolved the problem by using a special character to introduce ASCII constants.

SOME NOTEWORTHY ENHANCEMENTS

Let us turn now to the other six kinds of errors in the log.

C, Clean-ups

The stickiest issue in TEX has always been the treatment of blank spaces. Users tend to insert spaces in their computer files so that the files look nice, but document processors must also treat spaces as objects that appear in the final output. Therefore, when you see documents nowadays that have been prepared by systems other than TEX, you often find cases where double spaces appear incorrectly between words; and when you see documents prepared with TEX, you run into cases where a necessary space between words has disappeared. I kept searching for rules that would be simple enough to be easily learned, yet natural enough that they could be applied almost unconsciously. I finally concluded that no such rules existed, and I opted for the best compromise I could find.

Several of the log entries refer to the question of optional spaces after a macro definition. In no. 133, I decided to ignore a space that appears there; this was prompted by experiences recorded in my comments following nos. 115 and 119. But no. 133 caused a timing problem in no. 560, because the macro definition had not been fully processed when TEX wanted to check for the optional space; if the user invoked the macro immediately, instead of putting a space there, TEX was not ready to respond. Finally in no. 606 I came to the conclusion that TEX users will best be able to keep their sanity

if I do not ignore spaces after definitions; then dozens of similar-appearing cases all have consistent rules.

(See also no. 220, for space after $$; nos. 361, 708, 720 and 723, for space after constants; no. 440, for space after active characters; and no. 632, for space after '\\'.)

G, Generalizations

TEX continued to grow new capabilities as people would present me with new applications. When I could not handle the new problem nicely with the existing TEX, I would usually end up changing the system. (But I kept the changes minimal, because I always wanted to finish and get on with other things. More about that later.)

Such generalizations were often built incrementally on the shoulders of their predecessors. For example, the original TEX78 had \output and \mark and macro definitions, which scanned and remembered lists of tokens, but there was no good way to assign a list of tokens to a 'token list variable' without causing macro expansion. Then TEX82 added a feature called \everypar, which Arthur Keller had long been lobbying for. One day I noticed that I could solve a user's problem in a tricky way by temporarily using \everypar to store a list of tokens. This was quite different from the intended use of \everypar, of course; so I introduced a new primitive operation called \tokens for such purposes (no. 559). Later, \everypar spawned several descendants called \everymath and \everydisplay (no. 568), \everyhbox and \everyvbox (no. 649), \everyjob (no. 657), \everycr (no. 688). I eventually found applications where \tokens was not enough by itself and I needed to borrow one of the \every features temporarily to do some non-standard hackery. So I finally replaced \tokens by an array of 256 registers called \toks (no. 713), analogous to TEX's existing arrays of registers for integers, dimensions, boxes and glue. TEX82 also acquired the ability to make assignments between different kinds of token_list variables (no. 746). In such ways I tried to keep the design 'orthogonal' as the language grew.

Of course every language designer likes to keep a language simple by applying Occam's razor. I was pleased to discover early in 1977 that simple primitive operations involving boxes, glue and penalties could account for many of the fundamental operations of typesetting. This was a real unification of basic principles, and it turned out to be even better when I realized that the concepts of ordinary line-breaking applied also to tasks that seemed much harder.[13] But I also fooled myself into thinking that TEX had fewer primitives than it really did, by 'overloading' operations that were essentially independent and calling them single features.

For example, my original design of TEX78 would break paragraphs into lines by ignoring all lines whose badness exceeded 200. Later (no. 104) I made this threshold value user-settable by introducing a new primitive called \jpar. Setting \jpar=2 was something like setting \tolerance=200 in TEX82; but I also included a peculiar new convention: if \jpar was odd, the paragraphs would be set with ragged right margins, otherwise they would be justified to the full width!

Thus, in my attempt to minimize primitives, I had loaded two independent ideas onto a single parameter. I had also packed half a dozen different kinds of diagnostic output into a single number called \tracing (see no. 199), whose binary digits were examined individually when TEX was deciding whether to trace parts of its operations.

Then I began to see the need for more user-settable numbers, and I shuddered to think at the resultant multiplicity of new primitives. So I replaced both \jpar and

\tracing by a *single* primitive called \chpar (no. 244); one could now say, for example, \chpar1=2 instead of \jpar=2. This change gave me the courage to add new parameters for hyphen penalties, etc., and I even added a new parameter to control the raggedness of right margins (no. 334). Now the parity of \jpar was irrelevant; henceforth, the right margins could be either straight or ragged, or they could be produced using some smoothly varying compromise between those extremes—'one third of the way to full raggedness.'

My decision to introduce \chpar in TEX78 was not too bad, because TEX is a macro language and I could immediately define \jpar and \tracing as abbreviations for \chpar1 and \chpar2. But still, those arbitrary numerical codes were inelegant. TEX82 now has fifty different primitive operations that denote integer-valued parameters, each with standard (but user-changeable) names. The old \jpar has become \tolerance and \pretolerance. The old \tracing has been unbundled into \tracingparagraphs, \tracingpages, \tracingmacros, and half a dozen more, with separate parameters such as \showboxdepth to govern the amount of display.

I, Interactions

About 15 per cent of the errors in the TEX log have been classified type I. The main issue in such cases is to help users identify and recover from errors in their source programs, and this is always problematical because there are so many ways to make mistakes. 'When your error is due to misunderstanding rather than mistyping, . . . TEX can only explain what looks wrong from its own viewpoint; such an explanation is bound to be mysterious unless you understand the machine's attitude'.[15] Which you don't.

Still, I kept trying to make TEX respond more productively, and every such change was logged as an 'error' in my original design. The most memorable error of this type was probably no. 213, when I first realized how nice it would be if I could insert a token or two that TEX could read immediately, instead of aborting a run and starting from scratch. (This was soon followed by no. 242, when deletion of tokens was also allowed in response to an error message.) I would never have thought of these improvements if I had not participated in the implementation and testing of TEX, and I have often wished for similar features in the compilers I have used since. This one feature must have saved me hundreds of hours as a TEX user during recent years.

Another improvement in interaction did not occur to me until several months and several hundred pages of output later. Error no. 338 records the blessed day when I gave TEX the ability to track 'runaways', parts of the program that were being processed in the wrong mode because of missing right delimiters. (Further refinements to that change were logged as entry nos. 344, 426 and 793.) Without such provisions, errors that TEX could not have detected until long after their appearance would have been much harder to track down.

There was another significant improvement in interaction that never made it into my error log, because I included it in the original TEX82 without ever putting it into TEX78. This is the short_display procedure, for showing the contents of 'overfull boxes' and such things in an abbreviated form easily understood by novice users. The short_display idea was invented by Ralph Stromquist, who installed it in his early version of TEX at the University of Wisconsin.

P, Portability

The first changes of type P were simply enhancements to the comments in my SAIL program, but the advent of WEB made it possible for TEX to become truly independent of the machine and operating system it was being run on.

Change no. 633 is perhaps the most instructive class-P modification: I decided to guarantee compatibility between DEC-like systems (which break the source file into lines according to the appearance of ASCII carriage_return characters) and IBM-like systems (which have fixed-length source lines reminiscent of 80-column cards),* in the following way: whenever TEX reads a line of input, on any system, it automatically removes all blank spaces that appear at the right end. The presence or absence of such blanks therefore cannot influence the behaviour of TEX in any way. An ASCII file whose lines are at most 80 characters long (as defined by carriage returns, with or without blanks in front of those carriage returns) can be converted to a file of 80-character records that will produce identical results with TEX, simply by padding each line with blanks.

Change no. 791 carried no. 633 to its logical conclusion.

Q, Quality

From the beginning, I wanted TEX to produce documents of the highest possible typographical quality. The time had come when computer-produced output no longer needed to settle for being only 'pretty good'; I wanted to equal or exceed the quality of the best books ever printed by photographic methods.

As Kernighan and Cherry have said, 'The main difficulty is in finding the right numbers to use for esthetically pleasing positioning. . . . Much of this time has gone into two things—fine-tuning (what is the most esthetically pleasing space to use between the numerator and denominator of a fraction?), and changing things found deficient by our users (shouldn't a tilde be a delimiter?)'.[23]

I too had trouble with numerators and denominators: change no. 229 increased the amount of space surrounding the bar line in displayed fractions, and I should have made a similar change to fractions in text. (Page 68 of the new Volume 2 turned out to be extremely ugly because of badly spaced fractions.) TEX82 was able to improve the situation because of my experiences with TEX78, but even today I must take special precautions in my TEX documents to get certain square roots to look right.

THE EVOLUTION PROCESS AS A WHOLE

Looking now at the entire log of errors, I am struck by the fact that my attitude during those years was clearly far from ideal: my overriding goal was always to finish, to finish, to get this long-overdue project done so that I could resume work on other long-overdue projects. I never wanted to spend extra time studying alternatives for the best possible typesetting language; only rarely was I in a mood to consider any changes

*Paradoxically, DEC has also introduced the VMS operating system, which has fixed-length lines that can include troublesome carriage-returns. But that is another story.

to TEX whatsoever. I wanted TEX to produce the highest quality, sure, but I wanted to achieve that with the minimum amount of work on my part.

At the end of almost every day between 29 March 1978 and 29 March 1980, I felt that TEX78 was a complete system, containing no bugs and needing no further enhancements. At the end of almost every day since 9 September 1982, I have felt that TEX82 was a complete system, containing no bugs and needing no further enhancements. Each of the subsequent steps in the evolution of TEX has been viewed not as an evolutionary step towards a vague distant goal, but rather as the final evolutionary step towards the finally reached goal! Yet, over time, TEX has changed dramatically as a result of many such 'final steps'.

Was this horizon-limiting attitude harmful, or was it somehow a blessing in disguise? I am pleased to see that TEX actually kept getting simpler as it kept growing, because the new features blended with the old ones. I was constantly bombarded by ideas for extensions, and I was constantly turning a deaf ear to everything that did not fit well with TEX as I conceived it at the time. Thus TEX converged, rather than diverged, to its final form. By acting as an extremely conservative filter, and by believing that the system was always complete, I was perhaps able to save TEX from the 'creeping featurism'[24] that destroys systems whose users are allowed to introduce a patchwork of loosely connected ideas.

If I had time to spend another ten years developing a system with the same aims as TEX—if I were to start all over again from scratch, without any considerations of compatibility with existing systems—I could no doubt come up with something that is marginally better. But at the moment I cannot think of any big improvements. The best such system I can envision today would still look very much like TEX82; so I think this particular case study in program evolution has proved to be successful.

Of course I do not mean to imply that all problems of typography have been solved. Far from it! There still are countless important issues to be studied, relating especially to the many classes of documents that go far beyond what I ever intended TEX to handle.

CONCLUSIONS

My purpose in this paper has been to describe what I think are the most significant aspects of the experiences I had while developing TEX, basing this on a study of more than 800 errors that I noted down in log books over the years. I have tried to interpret many specific facts and observations in a sufficiently general way that readers may understand how to apply similar concepts to other software developments.

In Volume 1 of *The Art of Computer Programming*,[25] I wrote:

> Debugging is an art that needs much further study . . . The most effective debugging techniques seem to be those which are designed and built into the program itself . . . Another good debugging practice is to keep a record of every mistake that is made. Even though this will probably be quite embarrassing, such information is invaluable to anyone doing research on the debugging problem, and it will also help you learn how to reduce the number of future errors.

Well, I hope that my error log in the appendix below, especially the first 237 items (which relate specifically to debugging), will be useful somehow to people who study the debugging process.

But if you ask whether keeping such a log has helped me learn how to reduce the number of future errors, my answer has to be no. I kept a similar log for errors in METAFONT, and there was no perceivable reduction. I continue to make the same kinds of mistakes.

What have I really learned, then? I think I have learned, primarily, to have a better sense of balance and proportion. I now understand the complexities of a medium-size software system, and the ways in which it can be expected to evolve. I now understand that there are so many kinds of errors, we cannot stamp them out by systematically eliminating everything that might be 'considered harmful'. I now understand enough about my propensity to err that I can accept it as an act of life; I can now be convinced more easily of my fallacy when I have made a mistake. Indeed, I now strive energetically to find faults in my own work, even though it would be much easier to look for assurances that everything is OK. I now look forward to making (and correcting) hundreds of future errors as I write Volume 4 of *The Art of Computer Programming*.

ADDENDUM: FIFTEEN MONTHS MORE

As I mentioned above, I began to write this paper in May 1987, but I decided to wait before publication until more time had gone by. Then I could present a 'complete' and 'final' record of TEX's errors.

Now it is September 1988, and I have decided to bring this paper to a possibly premature conclusion, because I am scheduled to present it at a conference.[26] TEX still has not shown encouraging signs of becoming quiescent; indeed, sixteen more entries have entered the error log since May 1987, including three as recent as June 1988. Therefore it still is not the right moment to manufacture TEX on a chip!

All errors known to me as of 1 September 1988, are now included in the appendix to this paper; the total has now reached 865.* I plan to publish a brief note ten years from now, bringing the list to its absolutely final form.

I have been paying a reward to everyone who discovers new bugs in TEX, and doubling the amount every year. Last December I made two payments of $40.96 each, and my chequebook has been hit for five $81.92 payments in recent months. I am desperately hoping that this incentive to discover the final bugs will produce them before I am unable to pay the promised amount. (Surely in 1998 I won't be writing cheques for $83,886.08?)

As I expected, half of the most recent errors have fallen into the surprise (S) category—even though surprises, by definition, are unexpected. But one of the others (error no. 854) was perhaps the most surprising of all, because it was the result of a terrible algorithm by a person who certainly should have known better (me). I wanted to multiply the two's-complement fixed-point number

$$A = -16 + a_1 \times 2^{-4} + a_2 \times 2^{-12} + a_3 \times 2^{-20}, \qquad 0 \le a_i < 256$$

*Errors 866 and 867 were added after this paper was first submitted.

by the positive quantity $Z/2^{16}$, where Z is an integer, $2^{26} \le Z < 2^{27}$, obtaining an answer of the form $P/2^{16}$ where P is an integer, $|P| < 2^{31}$; all intermediate quantities in the calculation were required to be less than 2^{31} in absolute value. My program did this by computing

$$C \leftarrow 16 * Z;$$
$$Z \leftarrow Z \text{ div } 16;$$
$$P \leftarrow ((a_3 * Z) \text{ div } 256 + a_2 * Z) \text{ div } 256 + a_1 * Z - C;$$

I should rather have computed

$$Z \leftarrow Z \text{ div } 16;$$
$$P \leftarrow ((a_3 * Z) \text{ div } 256 + a_2 * Z) \text{ div } 256 + (a_1 - 256) * Z$$

(Consider, for example, the case $Z = 2^{26} + 15$ and $a_1 = a_2 = a_3 = 255$, so that $A = -2^{-20}$. The first method gives $P = -304$; the second method gives the correct answer, $P = -64$.)

Let me close by discussing one more recent error, no. 864. This change yields only a slight gain in efficiency, so I need not have made it; but it was easy to correct one more statement while I was fixing no. 863. It is an instructive example of how a design methodology based on invariants might not lead to the best algorithm unless we think a bit harder about what is going on.

Here is the idea: each run of TEX determines a threshold value θ above which the (one-word) charnodes will reside, below which all other (variable-size) nodes will be stored. Actually there are two values, θ_0 and θ_1; memory positions between θ_0 and θ_1 are empty. (In the program, θ_0 is actually called lo_mem_max, and θ_1 is called hi_mem_min.) TEX changes θ_0 and θ_1 conservatively as it runs, so that they will converge to values appropriate to particular applications. The boundary value θ was originally fixed at compile time; this transition to 'late binding' was change no. 819.

When TEX needs more space for charnodes, it usually sets $\theta_1 \leftarrow \theta_1 - 1$; when TEX needs more space for variable-size nodes, it usually sets $\theta_0 \leftarrow \theta_0 + 1000$. But we need to have $\theta_0 < \theta_1$. Therefore, instead of setting $\theta_0 \leftarrow \theta_0 + 1000$, my original code said

$$\text{if } \theta_1 - \theta_0 > 1000 \text{ then } \theta_0 \leftarrow \theta_0 + 1000$$
$$\text{else if } \theta_1 - \theta_0 > 2 \text{ then } \theta_0 \leftarrow (\theta_0 + \theta_1 + 2) \text{ div } 2$$
$$\text{else } \langle \text{Report memory overflow} \rangle.$$

(The variable θ_0 had to increase by at least 2.) Chris Thompson of Cambridge University pointed out that this strategy, although preserving the necessary invariants, is discontinuous. If $\theta_1 - \theta_0 = 1001$, the algorithm gobbles up all the discretionary space that is left. Therefore change no. 864 substituted better logic:

$$\text{if } \theta_1 - \theta_0 \ge 1998 \text{ then } \theta_0 \leftarrow \theta_0 + 1000$$
$$\text{else if } \theta_1 - \theta_2 > 2 \text{ then } \theta_0 \leftarrow \theta_0 + 1 + (\theta_1 - \theta_0) \text{ div } 2$$
$$\text{else } \langle \text{Report memory overflow} \rangle.$$

The new version also avoids problems on certain computers when θ_0 and θ_1 are negative; that was error no. 863. (Of course, when TEX is this close to running out of

memory, it probably will not survive much longer anyway. I am grasping at straws. But I might as well grasp intelligently.)

ACKNOWLEDGEMENTS

I have already mentioned that the TEX project has had hundreds of volunteers who helped to guide me through all these developments. Their names can be found in the rosters of the TEX Users Group; I couldn't possibly list them all here. Luis Trabb Pardo and David R. Fuchs were my 'right-hand men' for TEX78 and TEX82, respectively. The project received generous financial backing from several independent sources, notably the System Development Foundation, the U.S. National Science Foundation, and the Office of Naval Research. The material on which this report has been based is now housed in the Stanford University Archives; I wish to thank the archivist, Roxanne L. Nilan, for her friendly co-operation. The preparation of this paper has been supported by U.S. National Science Foundation grant CCR-86-10181. Thanks are due to the referee who helped me to remove errors not from TEX but from this paper. And above all, I want to thank my wife, Jill, for ten years of exceptional tolerance; software development is much more demanding than the other things I usually do. Jill also helped me to design the format for the appendix that follows.

APPENDIX: THE COMPLETE ERROR LOG

Each entry is numbered and cross-referenced (where possible) to other entries and to the TEX program, as explained in the text above. Sometimes I have given credit to the person who detected the error or suggested the change, but (alas) I did not always remember to note such information down. Here are the initials of people who made so many contributions that I have abbreviated their names in the log entries:

ARK	Arthur Keller
CET	Chris Thompson
DRF	David Fuchs
FY	Frank Yellin
HWT	Howard Trickey
JS	Jim Sterken
LL	Leslie Lamport
MDS	Mike Spivak

10 Mar 1978

1 Rename a few external variables to make their first six letters unique. L
2 Initialize *escape_char* to -1, not 0 [it will be set to the first character input]. §240 D
3 Fix bug: The test '*id* < *'200*' was supposed to distinguish one-letter identifiers from longer (packed) ones, but negative values of *id* also pass this test. §356 L
4 Fix bug: I wrote '**while** $\alpha \wedge (\beta \vee \gamma)$' when I meant '**while** $(\alpha \wedge \beta) \vee \gamma$'. §259 B
5 Initialize the input routines in INITEX [at this time a short, separate program not under user control], in case errors occur. §1337 R
6 Don't initialize *mem* in INITEX, it wastes time. §164 E
7 Change '*new_line*' [which denotes a lexical scanning state] to '*next_line*' [which denotes *carriage_return* and *line_feed*] in print commands. B
8 Include additional test '$mem[p] \neq 0 \wedge$' in *check_mem*. §168 F
9 Fix inconsistency between the *eq_level* conventions of *macro_def* and *eq_define*. §277 M

- *About six hours of debugging time today.*
- *INITEX appears to work, and the test routine got through start_input, chcode [the TEX78 command for assigning a cat_code], get_next, and back_input the first time.*

11 Mar 1978

10 Insert space before '(' on terminal when opening a new file. §537 I
11 Put '$p \leftarrow link(p)$' into the loop of *show_token_list*, so that it doesn't loop forever. §292 F
12 Shift the last item found by *scan_toks* into the *info* field. [With SAIL all packing of fields was done by arithmetic operations, not by the compiler.] §474 L
12 ↦ **13** Fix the previous bugfix: I shifted by the wrong amount. §474 B
14 Add a feature that prints a warning when the end of a file page occurs within a macro definition or call. [System dependent.] §336 I

- *Unintended bugs in my test routine [a format intended eventually to typeset The Art of Computer Programming] helped check out the error recovery mechanisms. For example, I had '`\lft{#}`' instead of '`\lft{##}`' inside a macro, and three cases of improper { and } nesting.*

15 Add the forgotten case '*set_font:*' to *eq_destroy*. §275 F
16 Change `\require` to `\input`. §376 C
17 Add code for the case *cur_cmd* = 0 [later known as the case '$t \geq cs\_token\_flag$'] when scanning a tokenlist. §357 F

- *That's the first "big" error I've spotted so far.*

18 Introduce a 'd' option in the error routine, to facilitate debugging. §84 I
19 Assign a floating-point constant *ignore_depth* to *prev_depth*, instead of assigning the integer constant *flag* [since *prev_depth* is type *real* in TEX78]. §215 L
20 Improve the readability and spacing of *show_node_list* output. §182, 187 I
21 Set the variable *v* before using the **case** construction in *show_node_list*, because there's one case where *v* didn't receive a value [as part of the field unpacking]. §182 F

- *About seven hours today.*

12 Mar 1978

- *One hour to enter yesterday's corrections and recompile.*
- *At this point TEX correctly located further unintended syntax errors in* `acphdr` *[the test file].*

22 Insert *debug_help* into *succumb*, giving a chance to look at memory before the system dies. §93 I
23 Use *eq_destroy* wherever necessary in *unsave*. §283 D

24 Change '$t \leftarrow (t-1) \bmod 8$' to '$t \leftarrow (t-1)$ land 7' in *id_name*, since SAIL has $-1 \bmod 8 = -1$. [At this time, *id_name* is a routine that unpacks control sequence names, according to a scheme that will become obsolete after change #422.] L

25 Remove the space that appears at end of paragraph. (I hadn't anticipated that.) §816 S

26 Throw away unwanted *line_feed* after getting a *carriage_return* in response to *in_chr_w* [a system routine for input from the terminal]. §83 L

27 Delete spurious call to *flush_list* in *end_token_list*. §324 B

- *Why did I make such a silly mistake?*

28 Fix bug in *get_x_token*: I forgot to say '*macro_call*' (which is the main point of that routine)! §380 F

- *While tracking that bug down, I found out incidentally that kerning is okay.*
- *Also TEX correctly caught an error* `0p` *for* `0pt`.

29 Fix bug in *scan_spec* (**while** instead of **repeat**). §404 L

30 Make the table entries for `\hfill` and `\hskip` consistent with the program conventions. §1058 M

31 Disable unforeseen coercion: When *scan_spec* put *hsize* on *save_stack*, the value changed from *real* to *integer*. §645 L

32 Use '`*`' instead of '`-1.0`' for running dimensions of rules in *show_node_list*. §176 I

33 Clear *mem*[*head*] to null in *push_nest* [in TEX82, this will be done by *get_avail*]. §216 D

- *A vrule link got clobbered because I forgot to do this.*

34 Translate ASCII control codes to special form when displaying them. §48, 68 I

- *Ligatures work, but show_node_list showed them funny.*

35 Remember to clear parameters off *save_stack* in *package* routine. §1086 F

- *About eight hours today.*

13 Mar 1978

36 Introduce a new variable *hang_first* [later the sign of *hang_after*]. §849 D

36 ↦ **37** Simplify the new code, realizing that if *hang_indent* = 0 then *hang_first* is irrelevant. §848 E

- *Time sharing is very slow today, so I'm mostly reading technical reports while waiting* **three hours** *for compiler, editor, and loading routine.*
- *I'm not counting this as debugging time!*
- *(Came back in the evening.)*

38 Spruce up the comments in the *line_break* routine, which appears to be almost working. §813 P

39 Rethink the setting of *best_line*; it's 1 too high in many cases. [The final line of a paragraph was handled in a treacherous way.] §874 D

40 Compute proper initialization for *prev_depth* when beginning an `\hbox` with a paragraph inside. [This refers to a special 'paragraph box' construction, used when an hbox of specified size becomes overfull; TEX78 doesn't have the concept of internal vertical mode.] §1083 D

41 Also initialize *tail* in that case. §1083 D

42 Also put the result of line-breaking into the correct list. M

43 Fix a typo in the *free_node* routine ('*link*' not '*llink*'); by strange chance it had been harmless until today. §130 T

44 Fix bug: *post_line_break* forgot to set *adjust_tail*. §889 F

45 Update *act_width* properly when looking for end of word while line breaking. §866 D

46 Repair the "tricky" part of *get_node*: I used the *info* field when I meant to say *llink*. §127 B

- *Now the* `\corners` *macro of* `acphdr` *works! [See* `\setcornerrules` *in The TEXbook, page 417.]*

47 Reset *contrib_tail* properly in *build_page*. §995 D
48 Fix typo (- for +) in computation of *page_total*. §1004 T
49 Change the page-breaking logic: TEX reached *fire_up* with *best_page_break* = *null* in one case, since the badness was too bad. §1005 S
50 Perform the operation *delete_token_ref*(*top_mark*) only when *top_mark* $\neq 0$. §1012 M
51 Make *scan_toks* omit the initial { of an `\output` routine. §473 F
52 Insert a comma to make memory usage statistics look better. §639 I

- *About seven good hours of debugging today.*
- *Tomorrow will be first-output day (I hope).*

14 Mar 1978

- *(Came in evening after sleeping most of day, to get computer at better time.)*
- *(Some day we will have personal computers and will live more normally.)*
- *8:30pm, began to enter corrections to yesterday's problems.*

53 Issue an error message for non-character in filename or in font name. §771 I
54 Display '...' for omitted stuff in *show_context* routine. §643 I
55 Watch out for the SAIL syntax '$\alpha + \beta$ lsh γ'; it doesn't shift $\alpha + \beta$ left (only β). §464 L

- *That error was very hard to track down; it created a spurious link field and sent hash*[0] = `\beta` *to the scanner!*
- *I could have found this bug an hour sooner if I had looked at the correct stack entries for name and token_type.*

56 Show the correct page number when tracing pages before output is shipped. §638 D
57 Remember to nullify a box after using it. §1079 F
58 Issue an error message if `\box255` isn't consumed by the output routine. §1015 I

- *I'm having trouble with the BAIL debugger; it makes an illegal memory reference and dies, when single-stepping past the entry to recursive procedures hlist_out and vlist_out. So I have to reload and be careful to go thru these procedures at high speed.*

59 Fix bug in comment (memory parameter description said $\geq$ not $\leq$). §11 P
60 Fix typo in definition of rule output (said x, y not $x0, y0$). [This part of the code went away when DVI files were introduced.] B
61 Correct the embarrassing bug in shellsort, where I said '$\leq$ *str*[*k*]' not '$\leq$ *t*'. [The first TEX had to sort all output by vertical position on page.] B
62 Make *start_input* set up *job_name* in the form needed by *shipout*; it uses obsolete conventions. §532, 537 M
63 Insert (and) into the SAIL macro definition of *new_string*. [This macro was for pre-DVI output.] L
64 Unscramble the parameters of *out_rule*: The declaration was $(x0, y0, x1, y1)$ while the call was $(x0, x1, y0, y1)$. M

- *4:30am, TEX's first page is successfully output!*
- *(It was '*`\titlepage\setcpage1\corners\eject\end`*'.)*

15 Mar 1978

- *10:30pm. Today I'm instrumenting the line-breaking routine and putting it through a bunch of tests.*
- *(The inserted instrumentation had bugs that won't be mentioned here.)*

65 Don't abort the job when *eq_destroy* redefines a TEX control sequence. §275 C

- *The first word of a paragraph won't be hyphenated ... so be it!*

66 Fix the typo in *line_break* that spoils the test for 'letters in the same font'. §896 T

- *The effect of that typo was to suppress all hyphenation attempts.*

25 ↦ **67** Replace the space at paragraph end by fillglue, not by zero. §816 B
68 Pack the hyphen character properly into its node. §582 L
69 Fix a typo ('←' for '+') in the computation of *break_width*. §838 T

70 Change the \end maneuver; the present code doesn't end the job, since I forgot that *back_input* uses *cur_tok*. §1054 M

71 Add a parameter to *try_break*, since the width is different at a discretionary hyphen. [This problem will be solved differently in TEX82, when discretionaries become much more general.] §840 A

72 Bypass kern nodes in pre-hyphenation. §896 F

73 Supply code for the forgotten case '< "a"' in pre-hyphenation. [This case was later generalized to a test of *lc_code*.] §897 F

74 Change $mem[q]$ to $prev\_break(q)$ in the reverse-linking loop. §878 B

- *(Such blunders. Am I getting feeble-minded?)*

75 Introduce special logic for *eject_penalty*; I was wrong to think that forced ejection was exactly like an infinitely negative penalty. §851 A

76 Use $(1+b)^2 - p^2$ when computing demerits with $p < 0$. §859 A

- *6:30am. The line-breaking algorithm appears to be working fine and efficiently. On small measures (about 20 characters per line), it gives overfull boxes instead of spaced out ones. Surprising but satisfactory.*

16 Mar 1978

- *9pm. The plan for tonight is to test page breaking and more paragraphing.*

77 Insert '\topskip' glue at beginning of page. §1001 G

78 Add '\pausing' feature. §363 I

79 Fix discrepancy: In *make_accent* I called *vpackage* with a pointer to the first list item, but *vpackage* itself assumes that the parameter is a pointer to that pointer. [The *vpackage* of TEX82 will be different.] §668 M

- *I checked for other lapses like that. Result: 14 calls OK, 12 NG.*

80 Create a new temporary list head location, *hold_head*, since there's a case where *vpackage* is improperly called with parameter *temp_head*. [At this time *vpackage* uses *temp_head* to make a list of all insertions found.] §1014 M

- *11:30pm. The machine is tied up again.*

81 Write code to handle charnodes in vlists; I forgot that I'd decided to allow them. [Later I prohibited them again!] §669 F

82 Combine the page lists before pruning off glue in *fire_up*; otherwise the pruning doesn't go far enough. §1017 F

25 ↦ **83** Fix typo where line-breaking starts: '*fill_glue*' should be '*new_glue*(*fill_glue*)'. §816 B

84 Add /q to xspool command (cosmetic change). [This changes a system command that causes TEX output to be printed on the Xerox Graphics Printer (XGP), the progenitor of future laser printers; the /q option says that the queue of printing requests should be displayed on the user's terminal.] §642 I

85 Don't write a form feed after the last page of output. §642 E

- *To fix this, I reorganized ship_out, and it became simpler.*

86 Correct a typo ('←' for '+') in the *vlist_node* case within *hlist_out*. [The output routines were quite different at this time, because output went directly to the XGP.] §622 T

87 Change the message 'completed page' to 'Completed for page'. §638 I

48 ↦ **88** Fix yet another typo in the computation of *page_total*: My original code said *stretch*(*p*) instead of *stretch*(*q*) (terrible). §1004 B

89 Document the dirty trick about *bot_mark*'s reference count. [That trick is, fortunately, no longer useful.] §1016 P

90 Rethink the algorithm for contributing an insertion: The original code tests for a page break after incrementing the totals but before the *contrib_list* is updated. [TEX78 handles insertions in a hardwired manner that will be greatly generalized in TEX82.] A

91 Fix *get_node* again: After the variable memory overflows, control falls through to *found* instead of going to the *overflow* call. §125 F

- *I spent several hours tracking down that data structure bug!*

92 Change *new_line* to *next_line* in yet another print command (see #7). B

75 ↦ **93** Amend the line-breaking algorithm: `\break` in paragraph doesn't work with really bad breaks. §851 A

- *A problem to be diagnosed tomorrow: Each time I run the test program, the amount of memory in use grows by 13 cells not returned.*
- *Seven hours tonight.*

17 Mar 1978

94 Introduce *dead_cycles* to keep `\end` active until *ship_out* occurs. §1054 G

95 Don't call *line_break* with an empty list. §1096 E

96 Take proper account of the (infinite) fillglue when computing the width of a paragraph line preceding a display. §1146 S

97 Add a new parameter to *hpack* so that *line_break* won't be called at the wrong time. [This is for the soon-to-be-obsolete feature described in #40.] M

98 Give a warning message if there's an `\hfill` in the middle of a paragraph; fillglue upsets the line breaker, because floating-point calculations don't have sufficient accuracy. §869 L

- *I spent an hour looking for another bug in TEX, but the following one was in METAFONT: The xgp_height data in fonts had been supplied wrong.*
- *It took two hours to recompile 32 fonts with proto-METAFONT.*

99 Make *show_node_list* and *show_token_list* more robust in the presence of software bugs. §182 R

97 ↦ **100** Do not remove nodes with *eject_penalty*, when the new parameter to *hpack* is *true*. D

97 ↦ **101** Put a fast exit into *hpack*; e.g., at glue nodes, test '**if** *paragraphing* ∧ ⟨current width is large⟩'. E

- *2am. I have to go to bed "early" tonight.*

18 Mar 1978

- *3:30pm. (Saturday)*

102 Add a parameter to *check_mem* (to suppress display unless needed). §167 I

103 Introduce a user-settable parameter `\maxdepth`, and pass it as a parameter to *vpackage*. §668 G

- *I realized the need for* `\maxdepth` *while fixing insertions (see #90).*

104 Introduce a user-settable parameter for *line_break*: The constant 2.0 in my original algorithm becomes `\jpar` [later `\tolerance`], to be set like `\tracing`. §828 G

105 Reclaim the *eject_penalty* nodes removed during line-breaking. §879 D

- *(Those were the 13 extra nodes reported on Thursday.)*
- *The init_align procedure worked right the first time!*
- *Also init_row, init_col. But then...*

106 Rethink the command codes: *endv* in a token list has too high a code for the assumptions of *get_next*. §207 S

107 Add a *prev_cmd* variable for processing delimited macro parameters; the original algorithm loses track of braces. [The rules will change slightly in TEX82, and *rbrace_ptr* will take on a similar function.] §400 A

108 Make the *get_next* routine intercept `&` and `\cr` tokens. §342 S

- *I'd thought I could just put* `&` *and* `\cr` *into big_switch [i.e., in the stomach of TEX, not the eyes]; that was a great big mistake.*

109 Make more error checks on *endv*; e.g., it must not occur in a macro definition or call. §780 R

108 ↦ **110** No, rethink alignments again; the new program still fails! §768 S

- *For the first time I can glimpse the hairiness of alignment in general (e.g., '`\halign{\u#\v&...`' when `\u` and `\v` are defined to include `&`'s and possible alignments themselves).*
- *I think there's a "simple" solution, by considering only whether an alignment is currently active [in §342].*
- *11:30pm. Went to bed.*

19 Mar 1978

- *Woke up with "better" idea on how to handle `&` and `\cr`.*
- *(Namely, to consider a special kind of `\def` whose parameters don't interrupt on `&`'s and `\cr`'s.)*
- *But replaced this by a much better idea (to introduce align_state).*
- *11pm. Began to use computer. Performed major surgery (inserting align_state and updating the associated routines and documentation).*

111 Pop the alignment stacks in *fin_align*. §800 D

110 ↦ **112** Fix a (newly inserted) typo in *show_context*. §314 T

110 ↦ **113** Set *align_state* false when a live `&` or `\cr` is found. [Originally *align_state* was of type *boolean*.] §789 D

114 Insert `\cr` when '}' occurs prematurely in an alignment. §1132 I

115 Remember to record *glue_stretch* when packaging an unset node. §796 M

- *I had a mistake in `acphdr` definition of `\quoteformat`; also extra spaces.*
- *My first test programs, used before today, were contrived to test macro expansion, line-breaking, and page layout.*
- *Next I'm using a test program based on Volume 2.*

116 Make carriage-return, space, and tab equivalent for macro matching. §348 C

117 Omit the reference count node when displaying a mark. §176 F

118 Correct a silly slip: I wrote '*type_displacement*' instead of '*value_displacement*' when packing data in a penalty node. §158 B

119 Don't go to *build_page* after seeing `\noindent`; TEX isn't ready for that. [In the original program, this was an instance of a bad **goto**.] §1091 M

- *I had undesired spaces coming thru the scanner in my macro definitions of `\tenpoint` [see The TEXbook, page 414].*
- *4am. TEX now knows enough to typeset page 1 of Volume 2!*
- *Also it did its first "math formula" (namely '`$X$`') without crucial error.*
- *(Except that the italic correction was missing for some reason.)*

120 Remember to decrement *cur_level* in *fin_align*. [The routines will eventually become more general and use *unsave* here.] §800 D

121 Remember to increment *cur_level* in error corrections by *handle_right_brace*. [A better procedure will be adopted later.] §1069 D

122 Fix a typo: ('{' instead of '}') in error message for *mmode* + *math_shift*. §1065 T

99 ↦ **123** Make *show_noad_list* more robust and more like the new *show_node_list*. [The routines will be combined in TEX82.] §690 R

124 Fix a typo in *char_box*: should say *font_info_real*. [In TEX78 a single array is used for both *real* and *integer*; in TEX82 things will be *scaled*.] §554 L

125 Fix typos in the definitions of *default_rule_thickness* and *big_op_spacing*; they shouldn't start at *mathex*(7). §701 B

126 Reverse the *before* and *after* conventions in math nodes. §1196 B

- *I had them backwards; this turned hyphenation on just before math, and off just after it!*
- *Seven and a half hours debugging today. Got through the test program a little more. But TEX blew up on '`$Y+1$`'; tomorrow I hope to find out why.*

20 Mar 1978

• *8pm. I decided to work next on a super-hairy formula.*

127 Change '`\ascii`' to '`\cc`' (character code). [This name will change again later, to '`\char`'.] §265 C

128 Don't bother to store a penalty node at the beginning of **$$** when the paragraph-so-far fits on a single line, since such a penalty has already been stored. [These conventions will change later, and the `\predisplaypenalty` will always be stored.] §1203 E

129 Avoid reference to *tail* in *build_page*, if *nest_ptr* > 0. §995 S

130 Correct a silly slip in *math_comp* (the exact opposite of what I did in #118). §1158 B

131 Rectify my mental lapse in *make_fraction*; I said *nucleus* instead of *thickness*. §743 B

132 Mask off the math class when scanning delimiters. §1160 F

133 Allow an optional space after `\def{...}`. [This decision will be retracted later.] §473 C

• *My test example is so complicated it causes the semantic stacks to overflow!*

134 Don't test for no pages output by looking at the channel status. §642 L

135 Fix typo in definition of `\mathop` (*open_noad* not *op_noad*). §1156 T

136 Rewrite *fin_mlist*, because '`\left(...\above...\right)`' doesn't parse correctly; the `\left` goes into the numerator, the `\right` into the denominator. §1184 A

137 Correct the use of *depth_threshold* in *print_subsidiary_data*: Simple fields get shown while others look empty. §692 B

78 ↦ **138** Return the carriage before showing the first line of a new file when pausing. §538 I

139 Fix bug: The call *show_noad_list*(*mem*[..]) should be *show_noad_list*(..), in the *incompleat_noad* case of *show_activities*. §219 M

• *3am. The whole messy formula has been parsed correctly into a tree.*

• *The easy part is done, now comes the harder part.*

140 Don't shift single characters down in *make_op*. §749 F

141 Make *clean_box* return a box (as its name implies), not an hlist. §720 D

• *Font info still isn't quite right, it has the wrong value of quad.*

142 Retain the italic correction when doing *rebox*; can make *glue_set* ≠ 0 a flag for this. [A better solution will be adopted later.] §715 A

143 Fix the bug that makes *rebox* bomb out: *value*(*p*) should be *value*(*mem*[*p*]). §715 B

• *6am; ten hours today. TEX didn't do* **$\pi\over2$** *correctly, but was close.*

• *I found that the rebox problem (#142) went away when I fixed the clean_box problem (#141); but I will leave the extra stuff about glue_set* ≠ 0 *in the program anyway, just for weird cases.*

144 Omit extra levels of boxing when possible in *clean_box*. §721 E

• *(To do this, I need to face the rebox problem anyway.)*

21 Mar 1978

• *10pm. The computer is rather heavily loaded tonight.*

145 Don't forget *thickness* when making a square-root sign (see #131). [The rule thickness will later be derived from the character height.] §737 F

146 Define *p* local to the *make_fraction* routine. §743 L

• *Unwittingly using the global p was a disaster.*

147 Don't show the amount of *glue_set* when it's zero. §186 I

142 ↦ **148** Make *glue_set* nonzero in the result of *var_delimiter*. §706 D

149 Fix bug: The *math_glue* function didn't return any result. §716 F

150 Fix typo in *char_box* (*c* not *w*); this caused a subscripted *P* to come out the same width as an unsubscripted *P*. [Later changes in the rules will move this computation to §755.] §709 T

144 ↦ **151** Revise *clean_box* to do operations that are needed often because of the *rebox* change. §720 P

152 Use the new *clean_box* to avoid a bug in `\sqrt{\raise...}`. §737 D

153 Change the definition of `\not` so that it's a relation (which will butt against the following relation). [All math symbols and Greek letters are defined in `INITEX` at this time, not in a changeable format definition.] Q

154 Give error message '`Large delimiter must be in mathex font`', instead of calling *confusion*, since the error can occur. [This particular error is impossible in TEX82.] §706 I

155 Change the use of p in *var_delimiter*; it isn't always set when I say **goto** *found*. §706 F

- *Another font problem now surfaced: The mathex meta-font didn't compute TEX info in a machine-independent way. (It took two hours to correct this.)*

156 Don't forget to set *type*(b) in all relevant cases of *var_delimiter*. §708 D

157 Use the correct sign convention for *shift_amount* in *hpackage*. §653 B

158 Always kern by *delta* when there's no superscript. §755 F

159 Declare *space_table* to be $[0..6, 0..6]$ not $[0..7, 0..7]$; otherwise its entries are preloaded into the wrong positions. [The *space_table* in TEX78 is 7×7; it will become 8×8 in TEX82, represented as a string called *math_spacing*.] §764 M

160 Use a negative value, not zero, to represent a null delimiter. [Actually zero will come back again later.] §685 C

127 ↦ **161** Change `\cc` to `\char`. §265 C

162 Don't use tricky subtraction on packed data when changing q to an *ord_noad* in *mlist_to_hlist*; subtraction isn't always safe. §729 L

163 Fix two typos in the *space_table* (`*` for `0`). §764 B

164 Initialize *cur_size* everywhere (I forgot it in two places). §703 F

165 Reset *op_noad* before resetting *bin_noad*. §728 A

166 Treat *display_style* + *cramped* the same as *display_style* inside *make_op*. §749 F

167 Shift the character correctly in the non-`\displaystyle` case of *make_op*. §749 B

- *Still another font problem: The italic corrections are wrong because the corresponding array was declared real in proto-METAFONT (and italic corrections were used in nonstandard way in* `mathex`*).*

168 Use *depth* instead of *height* in *var_delimiter*. [Later, both were used.] §714 B

169 Skew the accents according to the font *slant*. [Soon retracted.] §741 Q

- *At this point I think nearly all the math routines have been exercised.*
- *Tomorrow they should work!*
- *Eight hours debugging today.*

22 Mar 1978

- *(Wednesday, but actually Thursday: I began at midnight because I was proofreading a paper.)*
- *I checked out the font access tables, slowly (i.e., all the* `\mathcode` *and special-character name entries were catalogued).*

169 ↦ **170** Do **not** consider slants after all in the math accent routine, since slanted math letters are put differently into fonts. §741 Q

171 Don't use q for two different things simultaneously in *make_math_accent*. §738 B

172 Fix bug in *compact_list* (I forgot to advance the loop variable). [This procedure became unnecessary in TEX82.] F

173 Avoid conflict between *var_delimiter* and *mlist_to_hlist*, which want to use *temp_head* simultaneously. §713 M

174 Fix bad typo in *overbar* routine (b for p). §705 T

- *Finally TEX got to after_math after dealing with that hairy formula...*

175 Fix another bad typo: p for b this time. §1199 T

176 Insert more parentheses (twice) because of '`lsh`' precedence in `SAIL`. §1199 L

36 ↦ **177** Use the new hanging-indentation conventions when formatting displayed equations. §1199 M

178 Recompute penalties so that break is allowed after *punct_noad*s. §761 Q

179 Center the large delimiters vertically. §749 F
180 Round all rule sizes (up) before drawing them. §589 Q
181 Provide more space over x in $\sqrt{x}$, and more space atop vincula. §705, 737 Q
182 Make large delimiters large enough to cover formula height (important for subscripts, superscripts). §762 Q
183 Insert `/ntn=33` on XGP prompt message so that complex math won't blow the device driver. [See #84.] I
161 ↦ **184** Update the comment about the meaning of `\char`, since it can be used in math mode. §208 P

- *Six hours today.*

23 Mar 1978

- *11pm, Maundy Thursday.*

104 ↦ **185** Make `\tracing` and `\jpar` follow block structure. §283 C

- *It took me two hours to enter yesterday's corrections, because the changes were so numerous.*

186 Fix bad call on *begin_token_list* when marks are to be scanned. §396 M

- *Now the formula looks like it should, modulo problems in fonts.*

187 Prevent an exponent from going below baseline + xheight/4. §758 Q
188 Change *quad* to *math_quad* when finishing a display (several places). §1199 B
189 Don't use *append_to_vlist* when putting an `\eqno` box on a separate line, because the page shouldn't break at glue there. [Later, the *append* will be used but preceded by an infinite penalty.] §1205 A
190 Increase the input *stack_size*; TEX may need to back up a lot. §11 S
191 Don't assume that p always points to a glue node when a page is broken. §1017 A
192 Use *epsilon* in *scan_spec* (I had used a different small constant). [This was a kludge to avoid the extra parameter later called *exactly* or *additional*.] §645 A
193 Introduce a new procedure *scan_positive_length*, to prevent negative or zero lengths in *scan_rule_spec*. [This restrictive rule will be "overruled" later.] §463 R
194 Fix ridiculous bug in the leaders routine of *vlist_out*: I had the initialization **inside** the loop! §635 B
195 Eliminate confusion between the two temp variables named h; one is *real* and the other is *integer*. §629 L
196 Include forgotten case (*leader_node*) in *hlist_out*. [Type *leader_node* will be absorbed into *glue_node* in TEX82.] §622 F
197 Don't forget to compute *x0* in variable horizontal rules. §624 F

- *Seven and a half hours today.*
- *TEX seems to be ready to tackle my test file based on Volume 2.*

198 Calculate *y00* in horizontal rules as an integer number of pixels from the baseline, so that the baseline doesn't jump. §589 S

25 Mar 1978

- *2am Saturday. (Might as well drop Friday.)*

185 ↦ **199** Make *def_code* consistent with the new `\tracing` conventions. [Many tracing options are packed into a single parameter called `\tracing`.] §1233 D
185 ↦ **200** Don't allow users to change nonexistent things like `\catcode1000`. §1232 R
110 ↦ **201** Reset *align_state* at beginning of *init_align*. §774 F
202 Don't forget *scan_left_brace* after `\noalign`. §785 F
110 ↦ **203** Set *cur_cs* ← 0 in *get_next*, after `\cr` causes a switch to the $\langle v_j \rangle$ template. §342 F

- *Ouch, that was a big bad bug, which took me three hours to find (since I thought TEX's low-level scanning mechanism was working).*

- *Note to myself: I* **knew** *it would be cleaner to define get_next so that it sets cur_cs to zero every time it begins [i.e., in §341, where this change will in fact be made in TEX82]. But I had avoided this on grounds of efficiency in the inner loop. Well, now I have earned this tiny bit of efficiency.*

204 Prohibit the first word of an unavailable node from becoming negative. [The storage allocator of TEX78 uses a negative value to signify a node that is available, just as '*link* = *max_halfword*' will signal availability in TEX82.] §124 D

- *That was another bad one, it's not my night.*
- *At least I'm developing more subtle diagnostic techniques.*

205 Remember to un-negate the top *save_stack* entry when *handle_right_brace* finishes an *insert_group*. [This routine was completely revised in TEX82.] §1100 M

185 ↦ **206** Initialize `\jpar` [i.e., `\tolerance`]. §240 F

207 Correct the display of insertion nodes by *show_node_list*. §188 B

208 Prevent *show_token_list* from generating really long strings when in a loop. §292 R

209 Increase the reference count of *bot_mark* when *vpackage* finds it. [This was later the job of *fire_up*.] §1016 D

210 Remember that the tokenlist for a mark ends with a }. §1101 M

211 Don't let *vpackage* lose the top insert. (It fails when the very first item is a `\topinsert`.) §1014 D

212 And when that stupid code is corrected, make it handle insertions first-in-first-out. §1018 A

- *Seven hours today.*

26 Mar 1978

- *Easter Sunday, will work till sunrise.*

213 Add an '`i`' feature to the *error* recovery routine. §87 I

214 Include a prompt. §87 I

215 Ignore space after `\noalign{...}`. §1133 C

- *Otherwise, things are going well tonight; I'm finding more bugs in my test program than in TEX.*
- *The '`i`' feature is proving to be very helpful.*
- *I increased the size of mem (now lo_mem_max* = 3500, *mem_max* = 10000*).*
- *In fact I just needed to increase it again (now lo_mem_max* = 4500, *mem_max* = 11000*).*

216 Make `INITEX` output *mem_top* for consistency checking. §1307 R

217 Calculate the size of delimiters by considering the enclosed formula's distance from the axis, not from the baseline. §762 A

- *I'm having trouble with a* `SAIL` *compiler bug; I must rearrange the program, more or less at random, until it compiles correctly. I hope the bug isn't more severe than it appears.*

210 ↦ **218** Don't put a new group on *save_stack* if a null mark is expanded. [TEX82 will remove the '}' from the mark text.] §386 D

- *I had to redo the typewriter-style font since its width tables were wrong.*
- *And I increased low-memory size again to 5500, then 6500.*
- *Finally the entire test program was TEXed. Happy Easter! Six hours today.*

27 Mar 1978

- *Beginning at 2:30am.*

219 Move `\vcenter` processing to the first pass of *mlist_to_hlist*; otherwise the height, depth, subscripts, etc., are way off. §733 A

220 Omit space after closing `$$`. §1200 C

- *Spacing is wrong in the formula* $Y_1 + \cdots + Y_k$*; I have to rethink the use of three dots.*

221 Make conditional thin space available to user as \≤. [Later will retract this.] §226 G

222 Introduce \dispaskip and \dispbskip [later called \abovedisplayshortskip and \belowdisplayshortskip]. §226 Q

- *Reminder: I need to test line-breaking with embedded math formulas.*

223 Make sure that *interaction* ≠ *error_stop_mode* in the 'Whoa' error [*fatal_error*]. §93 I

224 Fix a big mistake in the *style_node* routine (which points to a glue spec, not to glue itself); somehow this didn't cause trouble yesterday. [In TEX78, style nodes double as placeholders for math glue like thin spaces.] §732 B

225 Make \fntfam obey group structure. [TEX78's \fntfam operation is a combination of TEX82's \textfont, \scriptfont, and \scriptscriptfont.] §1234 C

- *At this point the test routine for Volume 2 works perfectly.*
- *But I will change the page width in order to check harder cases.*

178 ↦ **226** Disable automatic line breaks after punctuation in math (e.g., consider $f(x, y)$). §761 Q

227 Represent italic corrections as boxes, not glue, so that they won't be broken. [The \kern command doesn't exist yet.] §1113 S

- *Eight hours today.*

228 Fix a bug that just clobbered the memory: Call *free_avail*, not *free_node*, in the *ins_node* case of *vpackage*. [This logic will change completely in TEX82.] §1019 B

29 Mar 1978

- *(Wednesday) Again beginning at 2:30am.*

229 Put still more space above and below fraction lines in displayed formulas. §746 Q

189 ↦ **230** Install an infinite penalty feature, which positively suppresses breaks; use it in displayed formulas whose \eqno doesn't fit. §1205 G

231 Call *build_page* after finishing a display; and don't go to the \noindent routine because of the next remark. §1200 F

232 Put \parskip glue just before a paragraph, not just after (since it interferes with a penalty after). §1091 S

- *Although the test program gives correct output, it generates 46 locations of variable-size memory and 280 of one-word memory that are not freed.*

233 Recycle the ulists and vlists in *fin_align*. §801 F

25 ↦ **234** Fix bug when deleting space at end of paragraph: *delete_glue_ref*(*cur_node*) not *delete_glue_ref*(*value*(*cur_node*)). §816 M

- *There's also a more mysterious type of uncollected garbage, a fraction_noad corresponding to* $p\choose$*, an incompleat_noad not completed.*
- *Couldn't find that one, so I recompiled with #233 and #234 corrected.*
- *Now it gains just 10 locations of variable-size memory and 7 of the other kind.*

235 Extend *search_mem* to search *eqtb* also. §255 I

143 ↦ **236** Fix bug in *rebox* when *list_ptr*(*b*) = 0. §715 D

- *The seven one-word nodes were generated by this bug; rebox put them onto a linked list starting with mem*[0]*, growing at the far end!*

237 Remember to complete each *incompleat_noad*. §1184 D

- *This solved the other mystery. I had never noticed that my test output was actually wrong:* $p\choose k$ *came out as 'k'.*
- *After these corrections, the test routine worked... I feel that TEX is now pretty well debugged (except perhaps for error recovery)—it's time to celebrate!*

1 Apr 1978

238 Don't quit after file lookup fails. §530 I

2 Apr 1978

239 Add *TEX_font_area*, so that it's easier to change the default library area associated with a device. §514 P

3 Apr 1978

240 Insert parentheses again, to cope with the precedence of `lsh` when packing data. (See #55 and #176.) §1114 L

• *I had never tried hmode + discretionary before!*

241 Remember that *back_error* requires *cur_tok* to be set. (Problem can arise during error recovery on parameter `#n` with n out of range.) §476 M

4 Apr 1978

242 Add a deletion feature to the *error* routine. §88 I

5 Apr 1978

243 Reset *space_factor* after \/ [this was later rescinded] and after math in text. §1196 Q

10 Apr 1978

104 ↦ **244** Replace `\jpar` and `\tracing` by a new primitive `\chpar` for parameters. It allows a user to change those quantities as well as the penalties for hyphens, relations, binary ops, widows. §209 G

14 May 1978

• *Beginning to typeset a real book (Volume 2, second edition), not just a test.*

245 Make math in text end with spacing as if it were followed by punctuation. [This rule will soon be rescinded.] §760 Q

246 Insert `\times` into the hash table; I left it out by mistake. [It will eventually move into `plain.tex`.] F

247 Change the names of Scandinavian accents from `\o`, `\oslash`, `\Oslash` to `\a`, `\o`, `\O`. [This will also move to `plain`.] C

17 May 1978

248 Fix a silly bug that hasn't been tweaked until today: '`\halign to size`' [obsolete in TEX82] used *vsize* instead of *hsize*. §645 B

19 May 1978

249 Add a `\topbaseline` feature [later called `\topskip`]. §1001 G

245 ↦ **250** Subtract the math spacing change of May 14. §760 Q

251 Skip past blanks in the *scan_math* procedure. [This blank-skipping will eventually go into *scan_left_brace*.] §403 A

252 Introduce a *missing_brace* routine [later generalized] to improve error recovery in *mmode* + *math_shift*, when the top of *save_stack* isn't a *math_shift_group*. §1065 I

253 Adjust the math spacing between closing parentheses and Ord, Op, Open, Punct. §764 Q

254 Make the underline go further under. §735 Q

96 ↦ **255** Compute the proper natural width when a displayed equation follows a paragraph whose fillglue has been deleted by *line_break*. §1146 S

20 May 1978

256 Fix the spurious value of *prev_depth* inside alignments. §775 A

257 Consider (and defeat) the following scenario: The u and v lists are built in *init_align* using *temp_head*; then while scanning '`\tabskip 2pt\rt{...}`' the macro `\rt` is expanded, clobbering *temp_head*. §779 S

• *That bug was more subtle than usual.*

258 Add the parameter *num3*, so that the positioning of `\atop` can be different from that for fractions. §700 Q

259 Add new parameters *delim1* and *delim2*, so that `\comb` can use fixed size delimiters, not computed as with `\left`. §748 Q

22 May 1978

221 ↦ **260** Change \≤ to \≥ and introduce \≤ as the negative of \≥. [Later obsolete.] §226 C

261 Fix the display of negative penalty nodes; *show_node_list* is confused when a negative value has been packed into the middle of a word. §194 L

- *Memory overflow just occurred with lo_mem_max = 7500 and mem_max = 16384. So I have to go to 15-bit pointers. (A problem on 32-bit machines?)*

23 May 1978

262 Add a new parameter *big_op_spacing5*, for extra space above and below limits of big displayed operators. §751 Q

263 Initialize *incompleat_noad* in `$$\halign{...}$$`. §775 F

- *That was another heretofore-untested operation. How much of the code has not yet been exercised?*

238 ↦ **264** Close the file when doing lookup-failure recovery. §27 F

265 Improve the error recovery for '`Extra &`'. §792 I

266 The top piece must be calculated mod 128 in *var_delimiter*, to guarantee a valid subscript range. [Obsolete in TEX82.] §546 R

252 ↦ **267** Fix a blunder in new *missing_brace* code. §1065 B

262 ↦ **268** Fix a blunder in new code for limits on display operators. §751 B

26 May 1978

269 Don't insert a new penalty after an explicit penalty in math mode. §767 Q

- *The hash table overflowed; I ought to make it much bigger.*

110 ↦ **270** Avoid possible bad memory references in alignment when there is erroneous input after `\cr`. [Instead of *extra_info*, the value of *cur_align* in TEX78 is negated, because we need only distinguish `\cr` from `&`.] §789 R

271 Make the dimension parameters like `\hsize` all global, so that they can be set in the `\output` routine. §279 S

- *This led to major simplifications, also to major surgery.*
- *[But it was a kludgy decision, overruled in TEX82.]*

94 ↦ **272** Don't forget to set the type of the new null box in the `\end` routine. §1054 D

27 May 1978

- *The data overflowed memory again, both low and high, doing Section 3.3.2.*

184 ↦ **273** Mask off extra bits of `\char` in math mode, to avoid bad memory references. §1151 R

274 Zero out the negative `\medmuskip` in script styles. §732 B

29 May 1978

275 Be prepared to handle an undefined control sequence during *get_x_token*. (Can fix this by brute force, using *get_token* instead of *get_next*.) §380 S

276 Correct the superscript shift when a single character is raised. §758 D

184 ↦ **277** Mask off all but 7 bits in `\char` routine, to avoid space-factor index out of range. §435 R

- *More memory capacity overflows.*

22 ↦ **278** Fix TEX's overflow stop so that I don't have to wait for loading of the `BAIL` debug routines. [System dependent.] §93 E

279 Remember to adjust the page number when a file page ends in mid-macro. [System dependent.] §306 F

5 Jun 1978

280 Make sure that the arguments of positioning commands don't overflow their field size. §610 R

281 Report the excess amount when giving an overfull box warning. §666, 677 I

7 Jun 1978

282 Use $\geq$ instead of $>$ as termination criterion in *var_delimiter*. §714 Q

283 Disallow `\eject` in math mode. [In TEX78, `\eject` is distinct from `\break`; in horizontal mode it includes TEX82's '`\vadjust{\break}`'.] §1102 R

284 Don't put too much clearance above `\sqrt` in text style. §737 Q

9 Jun 1978

110 ↦ **285** Make *align_state* an integer variable, not *boolean*, so that `\eqalign` can be within another `\eqalign`. §309 G

286 A `\mark` should expand its input. §1101 C

10 Jun 1978

287 Provide for preloading of fonts. §1320 E

288 Close the output file before switching to edit the input file with the '`e`' option. §84 L

289 Return adjustments found by *hpack* to free storage if they're not used. [Later, *hpack* will detach them only when they're used.] §655 E

290 Strive for consistency between *make_under* and *make_over*. §735 Q

18 Jun 1978

236 ↦ **291** Fix a serious error in *rebox* ('*b*' instead of '*list_ptr*(*b*)'). §715 B

• *Strange that such a bug would now surface for the first time!*

292 Remove `\deg` from `INITEX`, since macros suffice. C

293 Add an extra hyphenation penalty for two hyphenated lines in a row. §859 Q

19 Jun 1978

294 Introduce the '*no_new_control_sequence*' switch. Among other things, this will prevent an undefined control sequence following *scan_math* from clobbering the save stack. §259 S

20 Jun 1978

295 Change the badness test '*glue* ≤ 0.0' to '*glue* ≤ 0.0001'. [TEX82 will avoid such problems by calculating badness without floating point arithmetic.] §99 L

296 Force *badness* to be at most 10^{19}. §108 R

297 Add *end_template* for better error recovery in alignments. §375 I

287 ↦ **298** Make `INITEX` more like the real TEX; my simple scheme for font preloading was no good because it left thousands of 'dead' words in memory. §8 E

299 Economize disk space by using internal arrays in load modules that aren't being reinitialized. [System dependent.] E

300 Move the declaration of *mem* to the semantics module, so that the object code will be more efficient. [System dependent. The code of TEX78 was divided into separately compiled modules for syntax, semantics, output, extensions, and general organization.] E

21 Jun 1978

• *Today I'm working on the user manual.*

301 Disallow `\input` except in vertical mode. [I will change this in TEX82, treating `\input` as a case of expansion.] §378 C

302 Add error recovery for *endv* and *par_end* occurring in math mode. §1047 I

303 Generalize `\ifT` to `\if T`. §506 G

22 Jun 1978

304 Preload the `\bullet` [later done by `plain.tex`]. F

256 ↦ **305** Get the correct *prev_depth* at the beginning of an alignment. §775 D

306 Change `\eject` so that it ejects only once. §1000 C

14 Jul 1978

307 Look in standard area if a file isn't found in the user's area. §537 I

308 Echo all online inputs in the transcript file. §71 I

19 Jul 1978

309 Equalize spacing when only one of numerator/denominator is big. §745 Q

310 Prevent subscript from getting too high above baseline. §757 Q

311 Avoid infinite loop when stack overflows: *push_input* should say '**if** *input_ptr* $\geq$ *stack_size* $\wedge$ *interaction* = *error_stop_mode*'. §321 R

22 Jul 1978

312 Make \quad meaningful outside math mode. (All fonts must be generated again!) §558 C

313 Show the nesting level at the end of *show_activities*. [But I decided not to do this in TEX82.] §218 I

314 Put in \> [namely, \mskip\medmuskip; TEX78 already has \≥, for conditional \thinmuskip, as well as the negative amounts \<, \≤]. Change the name of vector accent from \> to \b. [Math spacing operators will become much more general in TEX82.] §716 C

25 Jul 1978

94 ↦ **315** Give the correct \hsize and \vsize to the null boxes created at \end. §1054 Q

94 ↦ **316** And don't "append" them. [Later this was changed, so that it would work better with generalized output routines.] §1054 A

297 ↦ **317** Remove the control sequence \endv, since error recovery is now better. §375 I

318 Define another mode of tracing: It says 'OK' and stops after \showlists. §1298 I

244 ↦ **319** Give better defaults to parameters. [Later done by plain.tex.] §209 Q

320 Allow more bits in the packed representation of \showboxdepth. §238 I

321 Scan past delimiters and/or dimensions when recovering from ambiguous fractions. §1183 I

322 Reduce accent numbers modulo 128 or 512, depending on the mode. §1165 R

323 Include a warning, '(\end occurred on level ...)'. §1335 I

28 Jul 1978

- *(I'm writing Chapter 27 of the manual: 'Recovery From Errors'.)*

324 Improve the error message in *scan_digit*. [This procedure will change its name to *scan_eight_bit_int*, when the number of registers increases from 10 to 256.] §433 I

325 Don't report overfull boxes if they're less than .1 point over. §666, 677 I

326 Give the user extra chances to define the font, if *read_font_info* is unsuccessful. §560 I

327 Change default recovery for bad parameter number from #1 to ##, since #1 won't always work and since ## is probably intended. §479 I

328 Omit the "Negative?" message on things like *scan_char_num*. §435 I

329 Improve error recovery when a large delimiter isn't in family 3. [Obsolete.] I

330 Give a more appropriate error message when the input is '$\right'. §1192 I

- *Currently TEX says 'Missing $'!*

331 Call *back_input* before the error message in *back_error*, not afterwards. §327 I

1 Aug 1978

332 Give an appropriate warning when there's no input file and the user types 'e'. §84 I

333 Increase the system pushdownlist size so that the manual will compile. [Procedures *hlist_out* and *vlist_out* can recurse deeply.] L

- *Yesterday I distributed 45 preliminary copies of the manual; today I took out the "debugging hooks" and put TEX up as a system program.*

2 Aug 1978

- *I'm typing Volume 2 again (currently in Section 4.2.2). Culture shock!*

334 Introduce a \ragged parameter, to indicate a degree of raggedness. [Previously, ragged-right setting was performed when the \tolerance/100 was odd! Eventually a better approach, with \rightskip and such things, will be discovered.] §886 G

335 Omit the 'widow penalty' in one-line paragraphs. §890 Q

5 Aug 1978

336 Generalize \pageno to \count⟨digit⟩. §236 G

285 ↦ **337** Update *align_state* when recovering from 'Missing {' and 'Extra }' errors. §1069, 1127 D

338 Show "runaway" tokens, making it easier to pinpoint an error. §306 I

22 Aug 1978

339 Add \predisplaypenalty. §1203 G

340 Clarify error messages; they should indicate when something has been inserted, etc. §1064 I

23 Aug 1978

114 ↦ **341** Substitute 'Extra }' for the losing 'Missing \cr' error message. §1069 I

213 ↦ **342** Go past online insertions in *show_context*. §311 I

343 Exact no penalty for breaking one line before a display. §1145 Q

338 ↦ **344** Check for runaways at end of file. §362 I

345 Give error message when a macro argument begins with }. §395 I

24 Aug 1978

213 ↦ **346** Remove extra line-feed in *show_context* after printing insertions. [System dependent.] §318 L

25 Aug 1978

347 Leave no glue at top of page, even after \eject. §997 Q

27 Aug 1978

348 Adopt Guy Steele's new version of the TEX source files. [He has recently made a copy and modified it by introducing compile-time switches for MIT conventions as an alternative to SUAI. This is the first time that TEX is being ported to another site; additional switches for PARC, TENEX, TOPS10, and TOPS20 will be added later, using the Steele style.] P

1 Sep 1978

349 Don't pass over leader nodes in the *try_break* background computation. [At this time, leaders have not yet been unified with glue.] §837 Q

82 ↦ **350** Prune away all penalties at the top of a page. §997 Q

4 Sep 1978

338 ↦ **351** Include '\' in error message about a runaway argument. §306 I

8 Sep 1978

• *I just remade all the fonts, with increased ligature field size.*

350 ↦ **352** Insert a necessary **goto** statement in the first branch of the new penalty routine within *build_page*. §997 B

30 Sep 1978

338 ↦ **353** Make the token list for runaway arguments meaningful outside of *macro_call*. (I just had a runaway argument ending with '\lcm', which turned out to be the control sequence in hashtable location 0.) §371 M

354 Avoid infinite loop when recovering from $$ in restricted horizontal mode. §1138 R

355 Fix two hyphenation bugs related to -ages, -ers. [A completely new algorithm for hyphenation will go into TEX82.] L

356 Add -est to hyphenation routine; also disable puz-zled and rat-tled, etc. Q

4 Oct 1978

357 Add new primitive \vtop. §1087 G

358 Treat implicit kerns properly after discretionary hyphens have been inserted. §914 Q

4 Nov 1978

359 Forget the half quad originally required at left and right when centering displayed equations without equation numbers. §1202 Q

11 Nov 1978

360 Don't let the postamble come out empty. [This could occur if no fonts were selected.] §642 R

15 Nov 1978

361 Allow optional space after digit in *scan_int* routine. §444 C

17 Nov 1978

362 Make the *check_mem* procedure slightly more robust. §167 R

20 Nov 1978

363 Make the `\par` in a `\def` match the `\par` that comes automatically with a blank line. (Suggested by Terry Winograd.) §351 C

364 Add new parameter `\mathsurround` for spacing before and after math in text. §1196 G

365 Extend `\advance` to allow increase by other than unity. [At this time it applies only to the ten `\count` registers, and it is called `\advcount`.] §1238 G

25 Nov 1978

366 Add a new primitive: `\unskip`. §1105 G

367 Add new primitives `\uppercase` and `\lowercase`. §1288 G

28 Nov 1978

338 ↦ **368** Don't let `\mark` and *macro_call* interfere with each other's *scanner_status*. §306 M

369 Omit extra } after *show_node_list* shows a `\mark`, since the right brace is already there. (See #210.) §176 M

370 Add a new primitive suggested by Terry Winograd: `\xdef`. §1218 G

29 Nov 1978

371 Delete a space following `\else{...}` also in the false case. [TEX78 uses braces, not `\fi`, for conditionals.] S

320 ↦ **372** Make `\tracing` set `\showboxbreadth` as advertised. §198 D

373 Account properly for kerns in width calculations of *line_break*. §866 F

364 ↦ **374** Delete a *math_node* at the beginning of a line. §148 Q

339 ↦ **375** Guarantee that `\predisplaypenalty=10000` will suppress page breaking before a display. §1005 A

6 Dec 1978

376 Change the file opening statement to allow lines up to 150 characters long. [System dependent.] L

16 Jan 1979

365 ↦ **377** Initialize *negative* properly in the `\advance` routine with a `\count` as argument. §440 F

20 Jan 1979

378 Try to keep complex, buggy preambles of alignments from crashing the program. §789 R

17 Feb 1979

376 ↦ **379** Give more detailed information when warning about a long line being broken. [System dependent; the buffer size in TEX78 is very limited.] I

380 Declare *p* local to *try_break*, for the "rare" case code. [My original program included the following comment: "This case can arise only in weird circumstances due to changing line lengths, and the code may in fact never be executed." Later Michael Plass will discover that variable line lengths require an entirely different algorithm, using *last_special_line*.] §847 L

334 ↦ **381** Don't omit the raggedness correction when the last line of paragraph has to shrink. [Obsolete in TEX82.] F

22 Feb 1979

363 ↦ **382** Don't forget to return from *get_x_token* after finding `\par`. §351 F

383 Add a new parameter: `\lineskiplimit`. §679 Q

384 Change the syntactic sugar: '`\hbox par`' replaces '`\hjust to ...{overfull}`'. [This vastly improves on the old idea (see #40), but there still is no internal vertical mode.] C

385 Introduce new names `\hbox` and `\vbox` for `\hjust` and `\vjust`. §1071 C

386 Add a new condition: `\ifpos`. [It will later be generalized to `\ifnum` and `\ifdim`.] §513 G

387 Add `vu` and `\varunit`. [TEX82 will eventually allow arbitrary internal dimensions as units of measure.] §453 G

312 ↦ **388** Add an `em` unit. §455 G

389 Legalize `\hbox spread` ⟨negative dimension⟩ [since *scan_spec* no longer uses the sign as a flag]. §645 C

10 Mar 1979

370 ↦ **390** Make *scan_toks* expand `\count` during `\xdef`. [This will change later when `\the` and `\number` are introduced.] §367 C

23 Mar 1979

391 Put only 100000 pt stretch at the end of a paragraph instead of 10000000000 pt. [In TEX78, "infinite" glue is actually finite but large; in the language of TEX82 we would say that `\parfillskip`, which is not yet user-settable, is being changed to be like `\hfil` instead of like `\hfill`.] §816 Q

392 Treat the last line of a paragraph more consistently with the other lines (e.g., when `\hfil` appears in mid-paragraph), by effectively inserting *inf_penalty* at the end. §816 Q

31 Mar 1979

393 Ensure that penalty nodes aren't wiped out, in weird cases where breaks occur at penalties that normally disappear. §879 S

27 Apr 1979

394 Correct the page number count when files begin with an empty page. [System dependent.] A

395 Allow the *math_code* table to be changeable via `\chcode`. [In TEX82, `\chcode` will split into `\mathcode` and `\catcode`.] §1232 G

332 ↦ **396** Don't accept '`e`' after an error message if not inputting from a file. §84 I

29 May 1979

397 Don't call *end_file_reading* if you haven't already invoked *begin_file_reading*; this could happen when trying to recover from an error in *start_input*. §537 F

7 Jun 1979

306 ↦ **398** Be sure to eject two pages, when `\eject` comes just at the time another break is preferable (e.g., when the page has just become too full). §1005 A

27 Jun 1979

354 ↦ **399** Don't say '`You can't do that in math mode`' when the user says '`$$`' in restricted horizontal mode! §1138 I

30 Jun 1979

400 Add `wd`, `dp`, `ht` dimension units. §455 G

307 ↦ **401** Don't try the system area for file names whose area is explicitly indicated. §537 I

1 Jul 1979

402 Allow letters as (ASCII) numbers [without the ' marker introduced later]. §442 G

2 Jul 1979

403 Fix a `\gdef` bug: If the control sequence was never defined before [this later became the *restore_zero* option], don't remove it at group end. §282 F

16 Jul 1979

320 ↦ **404** Update *show_noad_list* to be like *show_node_list*. [The two routines, originally separate, will be merged in TEX82.] §238 I

18 Jul 1979

405 Extend capacity from 32 fonts to 64 fonts if desired. §134 G

406 Add new *extra_space* parameter to all text fonts (requested by Frances Yao). §558 Q

407 Make each *node_noad* print properly in *show_noad_list*. §183 F

408 Make `\jpar` allow any break if it is 1000000 or more. [In TEX82, a `\tolerance` of 10000 or more allows any break.] §851 Q

23 Jul 1979

409 Introduce new primitives `\hfil`, `vfil`, `\hfilneg`, `\vfilneg`. §1058 E

410 Add `\ifmmode`. §501 G

411 Add `\firstmark`. §1012, 1016 G

412 Allow break at leaders (horizontal mode only). §149 C

25 Jul 1979

213 ↦ **413** Revise *error* so that online insertions work properly after end-of-file errors. §336 I

411 ↦ **414** Change '**if** *first_mark* $\neq 0$' to '**if** *first_mark* ≥ 0' [because -1 is used to indicate 'not yet given a value']. §1012 B

28 Jul 1979

370 ↦ **415** Stop `\xdef` from expanding control sequences after `\def`'s. [This decision will be rescinded later, after several more years of experience with macro expansion will suggest better ways to cure the problem.] §366 C

416 Change symbolic printout for control symbols. [System dependent.] §49 I

308 ↦ **417** Avoid linefeeds in the transcript file. [System dependent.] L

370 ↦ **418** Expand `topmark`, etc., in `\xdef`. §366 C

4 Aug 1979

413 ↦ **419** Fix an error introduced recently: `\par` was suddenly omitted at end of page. [System dependent.] B

11 Aug 1979

420 Change error messages that use `SAIL` characters not in standard ASCII. §360 P

28 Aug 1979

411 ↦ **421** Move the command '*first_mark* ← -1' from *vpackage* to *fire_up*. §1012 D

403 ↦ **422** Correct a serious `\gdef` bug: Control sequences don't obey a last-in-first-out discipline, so TEX loses things from the hash table when deleting a control sequence. §259 S

- *To fix this, I either need to restrict TEX (so that `\gdef` can be used inside a group only for control sequences already defined on the outer level) or need to change the hash table algorithm. Although all applications of TEX known to me will agree to the former restriction, I've chosen the latter alternative, because it gives me a chance to improve the language: Control sequences of arbitrary length will now be recognized.*

423 Make sure that *unsave* cannot call *eq_destroy* with a value from the upper part of *eqtb*. §282 D

- *I noticed this long-standing bug while fixing #422. It had very low probability of causing damage (e.g., it required a certain field of a floating-point number to have a certain value), but it would have been devastating on the day it first showed up!*

29 Aug 1979

424 Call *eq_destroy* when a control sequence is `\gdef`'ed after being `\def`'ed. §283 F

418 ↦ **425** Treat the first token consistently when `\topmark` and its cousins are expanded in *scan_toks*. §477 F

- *Now I've checked things pretty carefully and I think TEX is "fully debugged."*

25 Jan 1980

338 ↦ **426** Display runaway alignment preambles. §306 I

427 Introduce active characters (one-stroke control sequences). [I don't yet go all the way: The meanings of '`x`' and '`\x`' have to be identical.] §344 G

7 Feb 1980

314 ↦ **428** Fix a glaring omission: Op space `\>` was never implemented in math mode! §716 F

25 Feb 1980

429 Add a new dimension '`ex`' (for units of xheight). §455 G

3 Mar 1980

427 ↦ **430** Allow the control sequence `\:` to be redefined [it was the 'select font' operator]; this allows the character : to be active. [Obsolete.] C

23 Mar 1980

- *An extend-TEX-for-the-eighties party:*

431 Add a new `\copy` feature. §204 G

432 Add a new `\unbox` feature. §1110 G

433 Add a new `\open` feature [later `\openout`]. §1351 G

434 Add a new `\send` feature [later `\write`]. §1352. G

435 Add a new `\leqno` feature, requested by MDS. §1204 G

436 Add a new `\ifdimen` feature [later `\ifdim`]. §513 G

437 Make `\`⟨space⟩ in vertical mode begin a paragraph. §1090 C

438 Add a new `\font` feature [replacing the silly previous convention that a font must be defined when it is first selected]. §1256 G

439 Add new `\parval` and `\codeval` features [later `\the` ⟨whatever⟩]. §413 G

427 ↦ **440** Don't let active characters gobble the following space. §344 C

208 ↦ **441** Add a new parameter to govern amount of token list dumped. [Obsolete.] §295 G

442 Add a new `\linebreak` feature [later replaced by `\break`]. §831 G

25 Mar 1980

- *(Still working on the above, also thought of more.)*

443 Add a new `\mskip` feature. §716 G

444 Add a new `\newname` feature (soon changed to `\let`). §1221 G

430 ↦ **445** Allow any control sequence to be redefined. §275 G

446 Send the output to the user's current file area, even when input comes from elsewhere. §532 I

27 Mar 1980

447 Compute the xheight for accents in math mode from family 1, not family 3. [Obsolete.] Q

28 Mar 1980

448 Increase minimum clearance between subscript and superscript. §759 Q

29 Mar 1980

222 ↦ **449** When a display follows a display, the second should have the 'shortskip' glue. §1146 Q

4 Apr 1980

445 ↦ **450** Look at current token meanings when trying to recognize `\tabskip` in alignment preambles. §782 A

23 Apr 1980

451 Estimate the length of printed output, for the new priority feature on our XGP device driver. [System dependent.] I

434 ↦ **452** Break long `\send` lines into pieces so that the file can be read in again. [System dependent.] C

19 May 1980

182 ↦ **453** Don't make `\left` and `\right` delimiters too large; they need to be only 90% of the enclosed size. [This eventually became `\delimiterfactor`.] §762 Q

21 May 1980

454 Add a new `\pagebreak` feature [later `\vadjust{\break}`]. §655 G

13 Jun 1980

- *Today I'm beginning to overhaul the line-breaking routine, and I'll also install miscellaneous goodies.*

455 Allow a radical sign to be in different font positions. §737 G

456 Clear empty tokenlists off input stacks to allow deeper recursions (suggested by Jim Boyce's macros for chess positions). §325 E

457 Make `\spaceskip` and `\parfillskip` changeable. §1228 G

458 Add a new parameter `\rfudge` (per request of Zippel) [later `\mag`]. §288 G

459 Add a new parameter `\loose` [later `\looseness`]; now parameters are allowed to take negative values. §875 G

460 Remove the variable *just_par*. [Obsolete; it was the *real* equivalent of an *integer*]. E

14 Jun 1980

461 Install new line-breaking routines, including `\parshape`. (These major changes are introduced as Michael Plass and I write our article.) §813 Q

462 Add a new parameter `\exhyf` [later `\exhyphenpenalty`]. §870 G

16 Jun 1980

444 ↦ **463** Change conventions in *eqtb* so that glue is distinguishable from other equivalents. §275 S

444 ↦ **464** Don't expand `\b` in `\xdef{\d\b{...}}` after `\let\d=\def`. [Obsolete.] A

444 ↦ **465** Avoid creating dead storage when doing *unsave* in certain regions. §275 D

17 Jun 1980

466 Allow negative dimensions in rules. §138 C

19 Jun 1980

463 ↦ **467** Make the new test for glue at the outer level of *show_eqtb*. §252 B

27 Jun 1980

453 ↦ **468** Don't let `\left` and `\right` become too small for big matrices. [This eventually became `\delimitershortfall`.] §762 Q

3 Aug 1980

469 Don't move extra-wide, numbered equations flush left unless they begin with glue. §1202 Q

15 Sep 1980

461 ↦ **470** Say '$\geq fz$' instead of '$> fz$' in the pre-hyphenation routine; I'd forgotten my definition of *fz* [a variable used to test for a sequence of lowercase letters in the same font]. §897 M

395 ↦ **471** Check the range of the index in `\chcode` before saving the old value. §1232 R

18 Sep 1980

457 ↦ **472** Don't forget to increase the reference count to \parfillskip, or it will mysteriously vanish. §816 D

19 Sep 1980

412 ↦ **473** Make leaders break like glue in both horizontal and vertical modes. §149 C

364 ↦ **474** Make \mathsurround break properly at left and right end of lines. §879 Q

13 Oct 1980

461 ↦ **475** Remove spurious overfull boxes generated when the looseness criterion fails. [Obsolete.] I

461 ↦ **476** Redesign the iteration for looseness; breakpoints were not chosen optimally. §875 A

461 ↦ **477** Avoid storing a lot of breakpoints when they are dominated by others. §836 E

366 ↦ **478** Don't say '*cur_node*' when you mean '*mem*[*cur_node*]'. §1105 B

461 ↦ **479** Prefer the oldest break to the youngest break when two break nodes have the same total demerits. §836 Q

461 ↦ **480** Don't make badness too big for floating-point calculations, when forced to make an overfull box. [Obsolete.] L

10 Dec 1980

481 Make it impossible to get unmatched '}' in a delimited macro argument. §392 R

482 Add new \topsep and \botsep features. [These are TEX78's way to put space at the edge of inserts, replaced in TEX82 by the \skip register corresponding to an \insert class.] §1009 G

6 Jan 1981

483 Install new routines for reading the font metrics, using Ramshaw's TFM files instead of TFX files. §539 P

484 Abort after reporting 100 errors, if not pausing on errors. §82 I

485 Add new \spacefactor and \specskip and \skip primitives. [At this time we write '\specskip3=10pt' and '\skip3' for what will become '\skip3=10pt' and '\hskip\skip3' in TEX82.] §1060 G

366 ↦ **486** \unskip is now allowed in internal vertical mode. §1105 G

26 Jan 1981

482 ↦ **487** Don't say '*mem*[*q*]' when you mean '*q*'. (See #143 and #478.) §1009 B

27 Feb 1981

417 ↦ **488** Put some linefeeds back into the transcript file, in order to prevent overprinting in listings. [System dependent.] I

489 Add a new \dpenalty feature [later \postdisplaypenalty]. §1205 G

490 Add the dimension cc for European users. §458 G

491 Make *scan_keyword* match uppercase letters as alternatives to lowercase ones (suggested by Barbara Beeton's experiments with \uppercase). §407 C

492 Add nonstop mode so that overnight batch processing is possible. §73 I

2 Mar 1981

422 ↦ **493** Fix a still more serious \gdef bug: The generality of \gdef almost makes it a crime to forget **any** control sequence names, ever! (The previous bug was only the tip of an iceberg.) §259 S

494 Issue warning message at the end of a file page if nesting level isn't zero. [System dependent.] I

5 Mar 1981

495 Keep track of maximum memory usage, for statistical reporting. [Obsolete.] §125 I

350 ↦ **496** Prune away glue and penalties at top of page after marks, sends, inserts. §1000 Q

497 Allow \mark in horizontal mode. [Later it will be \vadjust{\mark...}.] §655 G

498 Allow optional space before a required left brace, e.g., `\if AA {...}`. [See #251.] §403 C

499 Issue an incomplete `\if` error, to help catch a bad `\if`. §336 I

17 Mar 1981

494 ↦ **500** Omit the warning message at end of a file page unless the nesting level has changed on that page. [System dependent.] I

310 ↦ **501** Fix the spacing when there is a very tall subscript with a superscript. §759 Q

20 Mar 1981

371 ↦ **502** Make space-eating after `\else` fully consistent between the true and false cases. [Obsolete.] S

24 Mar 1981

496 ↦ **503** Change *glue_spec_size* to *ins_spec_size* in *vpackage* [where insertions are done]. [Obsolete.] B

5 Apr 1981

501 ↦ **504** Fix a typo ('+' instead of '-') in the new subscript code; this shifted certain subscripts down instead of up. §759 B

18 Apr 1981

505 Make leaders with rules of specified size act like variable rules. §626, 635 G

29 Apr 1981

461 ↦ **506** Don't consider *badness* > *threshold* at a line `\break` except in an emergency. §854 A

13 Jul 1981

402 ↦ **507** Allow other characters as numbers. §442 C

294 ↦ **508** Avoid dead storage if a *no_new_control_sequence* error occurs. [Obsolete.] §259 R

509 Add a new `\ifx` feature. §507 G

510 Add new features `\xleaders` and `\cleaders`. §626, 635 G

14 Jul 1981

507 ↦ **511** Amend the new code for constants; the '.' in '.5' is thought to mean *'056*! §442 S

507 ↦ **512** And fix an egregious blunder in that code: New commands at the end of a procedure are ignored when earlier statements exit via **return**. §442 L

4 Aug 1981

513 Accept alphabetic codes for all online error recovery options, instead of insisting on control codes like line feed or form feed. [The original error-recovery codes were suggested by the conventions of the `SAIL` compiler.] §84 P

514 Add a new `\thebox` feature [later `\lastbox`]. §1079 G

7 Aug 1981

515 Add `fil`, `fill`, and `filll` as units for glue stretching or shrinking. §454 G

516 Suppress the overfull box error when shrinkage amount is negative. §664 I

9 Aug 1981

517 Let unset boxes inherit the size of their parent in alignments. §810 Q

12 Apr 1982

518 Make `INITEX` dump out the *font_dsize* array needed by the new `DVI` output module. §1322 F

1 May 1982

151 ↦ **519** Fix *clean_box* so that *mlist_to_hlist* cannot make $link(q) = 0$ and $type(q) = glue\_node$. §720 S

- *[That was the historic final change to TEX78. All subsequent entries in this log refer to TEX82.]*

28 Sep 1982

- *Here are the first changes made to the preliminary listing of TEX82 that was published by the TEX project earlier this month.*

520 Insert the missing cases *letter* and *other_char* after *x_token* looks ahead. §1036 F
521 Change '`\pause`' to '`\pausing`'. §236 C
522 Reset *overfull_rule* when determining tabskip glue. §804 D
523 Fix the logic for scanning `\ifcase` [in obsolete syntax—everything is still done with braces since '`\fi`' doesn't exist yet]. §509 A

30 Sep 1982

524 Change `"0.0"` to `"?.?"` (suggested by DRF). §186 I

2 Oct 1982

525 Use conditional thin spacing next to 'Inner' noads. §764 Q
526 Make thick spaces conditional. §766 Q

4 Oct 1982

527 Increase *trie_size* from 7000 to 8000, because of Frank Liang's improved (but longer) hyphenation patterns. §11 P

6 Oct 1982

528 Change the string lengths to match the new *TEX_format_default*. §520 F

- *Version 0 of TEX is being released today!*

8 Oct 1982

529 Fix a blunder: I decreased *h* mod a quarterword when it should have been decreased mod *trie_op_hash_size* (HWT). §944 B

9 Oct 1982

530 Fix a typo ('!' not '&') in the WEB documentation. §524 P
531 Remember to call *initialize* if a different format was preloaded (Max Díaz). §1337 F

- *Version 0.1 incorporates the above changes.*

12 Oct 1982

532 Add the '`\immediate`' feature, by popular request. §1375 G

- *Version 0.2 incorporates this (somewhat extensive) change.*

13 Oct 1982

533 Introduce new WEB macros so that *glue_ratio* is more easily changed. §109 P

- *I began writing The TEXbook today: edited the old preface and searched in the library for quotations.*

14 Oct 1982

534 Change the type of *hd* to *eight_bits*; it's not a *quarterword* (HWT). §649 B
535 Revise the optimization of DVI commands: It's not always safe to eliminate *pop* when the preceding byte is *push*, since DVI commands have variable length! (Embarrassing oversight caught by DRF.) §601 S

15 Oct 1982

536 Test '*prev_depth* > *ignore_depth*', not '$\neq$'. §679 C

- *Version 0.3 incorporates the above changes.*

16 Oct 1982

537 Omit definition of *align_size*; it's never used (Bill Scherlis). §11 P
538 Inhibit error messages when packaging box 255. §1017 I

21 Oct 1982

539 Subtract *width*(*q*) from *page_goal*, don't add it to *page_so_far*[1]. §1009 A

- *The comment in §982 is correct, and so was my first draft of this code; but when desk checking the program some months after writing it, I introduced this bug, believing that I was making the algorithm more elegant or something.*

- *Version 0.4 incorporates the above changes.*

22 Oct 1982

540 Increase the amount of lower (variable-size) memory from 12000 to 13000, since the TEX program listing now needs about 11500. [At this time there still is a fixed boundary between upper and lower memory.] §12 P

541 Add a new parameter `\boxmaxdepth`. §1086 G

- *Version 0.5 incorporates the above changes.*

26 Oct 1982

542 Fix an off-by-one error caught by Gabi Kuper and HWT. (I forgot '+ 1'). §1317 B

543 Fix the spacing of displayed control sequences: *print_cs* should base its decision on $cat\_code(p - single\_base)$, not $cat\_code(p)$. §262 B

- *The* `TRIP` *test detected this bug, but I didn't notice.*

27 Oct 1982

544 Set *math_type* before saying $fetch(nucleus(q))$, since fetching can have a side effect. §752 S

28 Oct 1982

545 Install a major change: Fonts now have identifiers instead of code letters. Eliminate the '`\:`' primitive, and give corresponding new features to '`\the`'. §209 G

- *Actually I began making these changes on October 26, but I needed two days to debug them and to put Humpty Dumpty together again.*
- *At this time I'm also drafting macros for typesetting The TEXbook.*
- *The above changes have been incorporated into Version 0.6.*

30 Oct 1982

- *After years of searching, I've finally found a definitive definition of the printer's point; and (unfortunately) my previous conjecture was wrong. The truth is that* 83 pc = 35 cm, *exactly; so I am changing TEX to conform.*

546 Revise unit definitions for the 'real' printer's point. §458, 617 C

- *Version 0.7 incorporates the above.*

1 Nov 1982

- *Oops! Retract error #546, and retract TEX Version 0.7; the source of my information about points was flaky after all. My original suppositions were correct, as confirmed by NBS Circular 570.*

4 Nov 1982

547 Revise the definition of `dd`, conforming to the definitive value shown me by Chuck Bigelow. §458 C

545 ↦ **548** Introduce "frozen" copies of font identifiers, to be returned by `\the\font`, so that font manipulation is more robust. §1257 R

5 Nov 1982

549 Reset *looseness* and paragraph shape when beginning a `\vbox`. §1083 D

6 Nov 1982

550 De-update *align_state* when braces are in constants. §442 D

551 Improve error recovery for bad alignments. §1127 I

- *Today I wrapped up Chapters 4 and 5.*

8 Nov 1982

552 Give more power to `\let`: the right-hand side needn't be a control sequence. §1221 G

553 Amend *show_context* to say '$(base\_ptr = input\_ptr) \vee$ '; otherwise undefined control sequences can be invisible in unusual cases (John Hobby). §312 I

554 Compute demerits more suitably by adding a penalty squared, instead of adding penalties before squaring. §859 A

- *Previously a slightly loose hyphenated line followed by a decent line was considered worse than a decent hyphenated line followed by a quite loose line.*

10 Nov 1982

555 Save a bit of buffer space by declaring *pool_file* only in `INITEX`. §50 E

11 Nov 1982

556 Introduce a new context indicator to clarify TEX's scanning state: A special type called *backed_up* is distinguished from other kinds of *inserted* lists; it is called 'recently read' or 'to be read again', while others are called 'inserted'. §314 I

557 Append a comment, '`treated as zero`', to the missing-number message. §446 I

558 Ignore the settings of `\hfuzz` or `\vfuzz` if `\hbadness` or `\vbadness` is less than 100. §666, 677 I

13 Nov 1982

- *Major surgery on the program is planned for today, because of new ideas suggested by correspondence with MDS and other macro writers.*

559 Introduce a new `\tokens` register; this will be useful and easy to add, since TEX already can handle `\everypar` and `\output`. §1227 G

560 Change *get_x_token* to *get_token* when scanning an optional space; then a construction like `\def\foo{...}\foo` won't complain that `\foo` is undefined. §443 C

- *This change was retracted when it was being debugged, because it could cause endv to abort the job. Then it was re-established again when I found that endv needed to be more robust anyway. [But it was eventually rescinded again.]*

561 Make `\span` mean 'expand' in a preamble. §782 G

562 Use three separate **if** tests instead of '∧' in the inner loop of *get_next*, to gain efficiency. §342 E

563 Introduce *get_r_token* so that assignments have uniform error messages and so that frozen equivalents cannot be changed. §1215 R

- *I gave a few variables more mnemonic names as I made these changes.*

564 Move conditional statements from the semantics ('stomach') part of TEX to the syntax ('mouth') part, by introducing '`\fi`'. Also introduce `\csname` and `\endcsname`. §372, 489–500 C

- *This makes macros much more predictable and logical, but it is by far the most drastic change ever made to TEX. The program began to come back to life only after three days of solid hacking.*
- *Several other things were cleaned up as part of this change because it is now more natural to handle them differently. For example, a null control sequence has now become more logical.*
- *The result of all this is called Version 0.8.*

18 Nov 1982

- *Today I resumed writing Chapter 8. Tomorrow I'm 2^{14} days old!*

21 Nov 1982

565 Declare *c* as a local variable for hyphenation (DRF). §912 F

566 Omit the "first pass" and try hyphenations immediately, if `\pretolerance` is negative (suggested by DRF). §863 E

567 Don't ship out incredibly huge pages; they might foul up `DVI` files. §641 R

2 Dec 1982

568 Add new features `\everymath` and `\everydisplay`. §1139, 1145 G

569 Add a new feature `\futurelet`. §1221 G

- *The changes above have been incorporated into Version 0.9 of TeX.*

7 Dec 1982

570 Add a new `\endinput` primitive (suggested by FY). §362, 378 G

8 Dec 1982

571 Try *off_save*, if `\par` occurs in restricted horizontal mode. (This avoids embarrassment if TEX says 'type a command or say `\end`', then when you type `\end` it says you can't!) [However, I soon retracted this change.] §1094 I

21 Dec 1982

572 Redefine `\relax` so that its *chr* field exceeds 127. (This facilitates the test for end in *scan_file_name*.) §265 A

566 ↦ **573** Call *begin_diagnostic* when omitting the first pass of line breaking. §863 F

574 Fix the logic of glue scanning: In `\hskip-1pt plus2pt` the minus should apply only to the `1pt`. §461 A

23 Dec 1982

575 Renumber the decimal codes in paragraph statistics for loose and tight lines; they were ordered backwards. §817 I

576 Treat a paragraph that ends with leaders like a paragraph that ends with glue. §816 C

577 Allow commas as alternates to radix points, for Europeans. §438 C

578 Change `\hangindent` to a normal dimension parameter. [It had been a combination of `\hangindent` and `\hangafter`, with special syntax.] §247 C

579 Make `\prevgraf` accessible to users. §422, 1244 G

580 Split `\clubpenalty` off from `\widowpenalty`. §890 G

- *I'm typing Chapter 14 while making these changes.*

24 Dec 1982

581 Use *back_input* instead of **goto** *reswitch* when inserting `\par`, because `\par` may have changed. §1095 S

25 Dec 1982

- *It's 10pm after a very Merry Christmas!*

582 Don't prompt for a new file name if `\openin` doesn't find a file. §1275 I

583 Add a new `\jobname` primitive. §472 G

584 Give the user a way to delete the dollar sign, when TEX decides to insert one. §1047 I

585 Allow optional equals after `\parshape`, and implement `\the\parshape`. §423, 1248 C

26 Dec 1982

586 Add an *if_line_field* to the condition stack entries, so that more informative error messages can be given. §489 I

549 ↦ **587** Introduce a *normal_paragraph* procedure, since initialization is needed also within `\insert`, `\vadjust`, `\valign`, `\output`. §1070 D

27 Dec 1982

588 Give users access to `\pagetotal` and `\pagegoal`. (Analogous to #679 and #585, but simpler.) §1245 G

589 Introduce `\tracingpages`, allowing users to see page-optimization calculations. Also split `\tracingparagraphs` off from `\tracingstats`. §987, 1005, 1011 I

- *The changes above have been incorporated into Version 0.91 of TEX.*

31 Dec 1982

590 Break the *build_page* procedure into two parts, by extracting the section now called *fire_up*. [This is necessary because some Pascal compilers, notably for IBM mainframes, cannot deal with large procedures.] §1012 P

564 ↦ **591** Make `\ifodd1\else` legal by introducing *if_code*. §489 S

592 Improve alignments when columns don't occur: Don't append null boxes for columns missing before `\cr`, and zero out the tabskip glue after nonpresent columns. §802 Q

593 Make the error message about overfull alignment more intelligible. §801, 804 I

- *The changes above have been incorporated into Version 0.92 of TEX82, which was the last version of 1982, completed at 11:59pm on December 31.*

3 Jan 1983

- *Today I'm beginning to write Chapter 15, and planning the* `\output` *routine of* `plain.tex`.

594 Change the logic of *its_all_over*; use *max_dead_cycles* instead of the fixed constant 100. §1054 C

595 Don't forget to *pop_nest* when an insert is empty. Also disallow optional space after `\insert`*n*`{...}`. §1100 F

4 Jan 1983

541 ↦ **596** Use the `\boxmaxdepth` that's declared inside a `\vbox` when packaging it. §1086 C

597 Rename `\groupbegin` and `\groupend` as `\begingroup` and `\endgroup`. §265 C

598 Make `\deadcycles` accessible to users. §1246 G

599 Base the split insertions on natural height plus depth, not on *delta*. §1010 Q

- *The changes above have been incorporated into Version 0.93.*

6 Jan 1983

600 Add *push_math* to handle a case where I forgot to clear *incompleat_noad*. (This long-standing bug was unearthed today by Phyllis Winkler.) §1136 D

588 ↦ **601** Add `\pageshrink`, etc., too. §1245 G

602 Introduce new parameters `\floatingpenalty`, `\insertpenalties`. Also adopt a new internal representation of insertion nodes, so that `\floatingpenalty`, `\splittopskip` and `\splitmaxdepth` can be stored with each insertion. §140, 1008 G

7 Jan 1983

603 Improve the rules for entering *new_line*, in particular when the end-of-line character is active. §343 Q

9 Jan 1983

604 Distinguish between implicit and explicit kerns. §155, 896 Q

605 Change the name `\ignorespace` to `\ignorespaces`. §265 C

560 ↦ **606** Don't omit a blank space after `\def`, `\message`, `\mark`, etc.; the previous hodgepodge of rules is impossible to learn. §473 C

- *The above changes appear in Version 0.94.*

12 Jan 1983

- *Beginning to write the chapters on math today.*

607 Add a new feature: active characters in math mode. §1151 G

15 Jan 1983

608 Fix a surprise bug: '`$1-$`' treated the - as binary. §729 A

609 Initialize *space_factor* inside discretionaries. §1117 D

16 Jan 1983

610 Fix an incredibly embarrassing bug: I forgot to update *spotless* in the *error* routine! F

- *While fixing this, I decided to change spotless to a more general history variable, as suggested by IBMers who want a return code.* §76, 82, 1335

611 Replace two calls of *confusion* by attempts at error recovery, in places where '`This can't happen`' could actually happen. §1027, 1372 I

18 Jan 1983

612 Introduce the *normalize_selector* routine to protect against startup anomalies when the transcript file isn't open. Also make *open_log_file* terminate in some cases. §92, 535 R

591 ↦ **613** Insert `\relax`, not a blank space, to cure infinite loop like `\ifeof\fi` (LL). §510 R

614 Change the old `\limitswitch` to `\limits`, `\nolimits`, and `\displaylimits`. Incidentally, this fixes a bug in the former positioning of integral signs. §682, 749 G

615 Give a `\char` in math mode its inherited `\mathcode`. §1151 C

525 ↦ **616** Make underline, overline, radical, vcenter, accent noads and `{...}` all revert to type Ord instead of type Inner. Introduce a new primitive `\mathinner`. (This fixes the spacing, which got worse in some ways after change #525.) §761 Q

- *I'm working on Appendix G today.*

19 Jan 1983

617 Introduce a `\mathchoice` primitive. §1174 G

618 Move `\input` from the stomach to the eyes. §378 C

619 Introduce `\chardef`, analogous to `\mathchardef`. §1036, 1224 C

620 Change `\unbox` to `\unhbox` and `\unvbox`; also add `\unhcopy`. §1110 G

621 Consider `\spacefactor`, `\pagetotal`, etc., as part of *prefixed_command*, even though they are always global. §1211 C

20 Jan 1983

622 Switch modes when `\hrule` occurs in horizontal mode or `\vrule` in vertical. §1090, 1094 C

623 Add a new `\globaldefs` feature. §1211 G

21 Jan 1983

624 Optimize the code, in places where it's important (based on frequency counts of TEX usage accumulated during the past week): Introduce *fast_get_avail* and *fast_store_new_token*; reduce procedure-call overhead in *begin_token_list*, *end_token_list*, *back_input*, *flush_node_list*; change some tests from '**if** $a \wedge b$' to '**if** a **then if** b'. §122, 371 E

22 Jan 1983

625 Save space in math lists: Don't insert penalties within restricted horizontal mode; simplify trivial boxes. §721, 1196 E

626 Fix a surprising oversight in the *rebox* routine: Ensure that b isn't a vbox. §715 S

545 ↦ **627** Make `\nullfont` a primitive, so that *cur_font* always has a value. (This is a dramatic improvement to TEX78, where a missing font was a fatal error called '`Whoa`'!) §552 C

24 Jan 1983

586 ↦ **628** List all incomplete `\if`'s when the job ends. §1335 I

29 Jan 1983

629 Change initialization of *align_state* so that `\halign\bgroup` works. §777 C

30 Jan 1983

625 ↦ **630** Be sure to test '*is_char_node*(q)' when checking for a trivial box. §721 D

- *By extraordinary coincidence, this bug was caught when somebody used font number 11 (= kern_node) in the second character of a list of length 2!*

631 Improve format for stats at end of run, as suggested by DRF. §1334 I

- *The changes above have been incorporated into Version 0.95.*

632 Don't ignore the space after a control symbol (except '`\ `'). §354 C

633 Remove all trailing spaces at the right of input lines, so that there's perfect compatibility with IBM systems that extend short lines with spaces. §31 P

3 Feb 1983

634 Assume that a *math_accent* was intended, after giving an error message in the case *mmode* + *accent*. §1165 I

635 Add new primitives `\iftrue` and `\iffalse`. §488 G

6 Feb 1983

636 Improve the accuracy of fixed-point arithmetic when calculating sizes for `\left` and `\right`. (I had started by dividing *delimiter_factor*, not *delta1*, by 500.) §762 A

12 Feb 1983

637 Change the name `\delimiterlimit` to `\delimitershortfall`. §248 C

638 Make `\abovewithdelims..` equivalent to `\above`; change the order of operands so that delimiters precede the dimension. §1182 C

607 ↦ **639** Remove the kludgy math codes introduced earlier; make `\fam` a normal integer parameter and allow `\mathcode` to equal 2^{15}. §1233 C

640 Don't let `\spacefactor` become 2^{15} or more. §1233, 1243 R

- *I finished drafting Chapter 17 today.*

14 Feb 1983

639 ↦ **641** Replace octal output (*print_octal*) by hexadecimal (*print_hex*) so that math codes are clearer. §67 I

619 ↦ **642** Don't forget *char_given* in the *math_accent* routine. §1124 F

17 Feb 1983

643 Switch modes when `\halign` occurs in horizontal mode, or `\valign` in vertical mode. §1090, 1094 C

18 Feb 1983

644 Add a new feature `\tracingrestores`. This requires a new procedure called *show_eqtb*, whose code can be interspersed with the *eqtb* definitions. §252 I

25 Feb 1983

622 ↦ **645** Suggest using `\leaders` when the user tries a horizontal rule in restricted horizontal mode. §1095 I

27 Feb 1983

646 Specify the range of source lines, when giving warning messages for underfull or overfull boxes in alignments. §662, 675 I

- *Why did it take me all day to type the middle part of Chapter 18?*

4 Mar 1983

647 Introduce a new feature `\xcr` (suggested by LL). [Changed later to '`\crcr`'.] §785 G

631 ↦ **648** Subtract out TEX's own string requirements from the stats. §1334 I

6 Mar 1983

649 Add new features `\everyhbox` and `\everyvbox`. §1083, 1167 G

9 Mar 1983

650 Avoid accessing *math_quad* when the symbol fonts aren't known to be present. §1199 R

533 ↦ **651** Introduce *float* and *unfloat* macros to aid portability (HWT). §109 P

652 Introduce new names `\abovedisplayskip` and `\belowdisplayskip` for the old `\dispskip`; also `\abovedisplayshortskip` and `\belowdisplayshortskip` for the old `\dispaskip` and `\dispbskip`. §226 C

10 Mar 1983

653 Unbundle `\romannumeral` from `\number` (suggested by FY). §468 C

12 Mar 1983

654 Ignore leading spaces in *scan_keyword*. §407 C

14 Mar 1983

631 ↦ **655** Use *write* and *write_ln* directly when printing stats. §1334 E

16 Mar 1983

602 ↦ **656** Refine the page-break cost function (introducing '*deplorable*', which is not quite '*awful_bad*'), after suggestion by LL. §974, 1005 Q

• *The changes above have been incorporated into Version 0.96.*

18 Mar 1983

657 Add a new feature `\everyjob` suggested by FY. §1030 G

19 Mar 1983

658 Don't treat left braces specially when showing macros. §294 I

659 Ignore blanks that would otherwise become undelimited arguments. §393 C

21 Mar 1983

660 Make `\lastskip` handle *mu_glue* as well as ordinary glue. §424 F

561 ↦ **661** Expand only one level in a preamble `\span`. §782 C

22 Mar 1983

662 Let a single `#` suffice in `\tokens`, `\message`, etc. (The previous rule, in which `##` was always required as in macros, was a loser especially in `\write` where you had to say `####`!) §477 C

663 Require the keyword '`to`' in `\read`. (This will avoid the common error of an incomplete constant when no space appears before the `\cs`.) Also allow terminal I/O as a default when a stream number is out of range. §482, 1225, 1370 C

26 Mar 1983

664 Replace `\ifeven`⟨countnumber⟩ by `\ifodd`⟨number⟩, for better consistency of language. §504 C

564 ↦ **665** Introduce the *change_if_limit*, to overcome a big surprise bug relating to `\if\if aabc\fi`. §497 S

• *Such examples show that cur_if might not be current, in my original implementation.*

28 Mar 1983

666 Tolerate non-characters as arguments to `\if` and `\ifcat`. §506 G

667 Change '`absent`' to '`void`', a better word. §487 C

668 Clear the *shift_amount* in `\lastbox`, since I don't want to figure out what it means in all cases. §1081 C

29 Mar 1983

669 Wake up the terminal before giving an error message. (This means a special *print_err* procedure is introduced.) (Suggested by DRF.) §34, 73 I

1 Apr 1983

• *Today I finished Chapter 21 (boxes) and began to draft Chapter 22 (alignments).*

670 Allow periodic preambles in alignments. §793 G

671 Make `\leaders` line up according to the smallest enclosing box. §627, 636 C

672 Allow hyphenation after whatsits (e.g., after items for an index). §896 Q

2 Apr 1983

673 Call *build_page* when `\par` occurs in vertical mode. §1094 Q

674 Clear *aux* in *init_row*, for tidyness. §786 C

4 Apr 1983

675 Let digits switch families in math mode. §232 C

7 Apr 1983

602 ↦ **676** Refine the test for not splitting an insertion. §1008 Q

8 Apr 1983

647 ↦ **677** Rename `\xcr` as `\crcr`, at LL's request. §780 C

9 Apr 1983

- *Took a day off and had a chance to help print a sample page on a 150-year-old letterpress in Murphys, California.*

11 Apr 1983

678 Recover more sensibly after a runaway preamble. §339 I

12 Apr 1983

679 Make `\read` span several input lines, if necessary to get balanced braces. §482 C

14 Apr 1983

680 Fix a subtle bug found by JS: §882 can make q a *char_node*, so we need to test '**if** $\neg$*is_char_node*(q)'. [Actually I discovered much later that the real bug was to omit '**else**' at this point.] §881 S

15 Apr 1983

681 Make `\uppercase` and `\lowercase` apply to all characters, regardless of category. §1289 C

- *7:30am. After working all night, I completed a draft of the manual thru Chapter 22, for distribution to volunteer readers.*
- *5pm. The changes above have been incorporated into Version 0.97.*

17 Apr 1983

682 Change '*small_number*' to '0..65' in the hyphenation routine (DRF). §901 R

683 Flush patterns in the input when the user tries `\patterns` outside of `INITEX` (suggested by DRF). §1252 I

- *Tomorrow I fly to England, where I'll lecture and write a paper about 'Literate Programming' [Comp. J. **27** (1984), 97–111].*

14 May 1983

663 $\mapsto$ **684** Improve the behavior of `\read` from terminal (suggested by Todd Allen at Yale). [I'd forgotten to implement the extended stream numbers in #663. Also, the prompt is now omitted if $n < 0$.] §484 I

18 May 1983

685 Restrict `\write` n to the transcript file only, if $n < 0$. §1350 I

686 Unify the syntax for registers and internal quantities. (Remove primitives called '`\insthe`' and '`\minusthe`'; rename *scan_the* to *scan_something_internal*, and change its interface accordingly; clean up command codes generally.) §209, 413 C

687 Introduce new parameters `\hoffset`, `\voffset`. §617 G

24 May 1983

688 Introduce a new parameter `\everycr` (suggested by MDS). §774, 799 G

- *Many macro writers and preliminary-manual readers have been requesting new features; I'll try to keep the language as concise and consistent as possible.*

25 May 1983

689 Introduce `\countdef`, `\dimendef`, etc. (suggested by DRF long ago, easy now in view of #686). §1224 G

690 Introduce `\advance`, `\multiply`, `\divide` (suggested by FY). §1240 G

691 Introduce `\hyphenchar`; this requires a new command *assign_font_int*, plus minor changes to about 15 modules. §915 G

692 Introduce `\skewchar` (easy because of #691). §741 G

693 Introduce `\noexpand`. (I had difficulty thinking of how to implement this one!) §358, 369 G

694 Introduce `\meaning`. §296 G

695 Remove '`dm`' and '`vu`'; allow the more general '`.5\hsize`'. §455 G

696 Change '`\texinfo` f n' to '`\fontdimen` n f'. §578 C

27 May 1983

697 Add a new feature `\afterassignment` (suggested by ARK). §1269 G

698 Adjust the timing so that commands like '`\chardef\xx=5\xx`' behave sensibly. §1224 C

28 May 1983

699 Ignore '`\relax`' as if it were a space, in math mode and in a few other places where `\relax` would otherwise be erroneous. §404 C

700 Improve `\mathaccent` spacing with respect to subscripts and superscripts (suggested by HWT). §742 Q

30 May 1983

594 ↦ **701** Terminate a job only when *dead_cycles* = 0. §1054 C

- *The changes above constitute Version 0.98.*

3 Jun 1983

- *I finished the draft of Chapter 23 (output routines) today.*

702 Allow `\mark` and `\insert` and `\vadjust` in restricted horizontal mode, and also in math mode. (This is a comparatively big change, triggered by the fact that `\mark` in a display presently causes TEX to crash with '`This can't happen`'!) The global variable *adjust_tail* is introduced. §796, 888, 1085 G

6 Jun 1983

695 ↦ **703** Replace (and generalize) the previous uses of `ht`, `wd`, and `dp` in dimensions by introducing the new control sequences `\ht`, `\wd`, and `\dp`. §1247 G

704 Display sub-parts of noads with the symbols ^ and _ instead of (and [. §696 I

694 ↦ **705** Allow `A..F` in hex constants to be *other_char* as well as *letter*. §445 C

7 Jun 1983

654 ↦ **706** Remove an instance of ⟨Scan optional space⟩, since it's now redundant. §457 E

707 Legalize `\mkern\thinmuskip` and `\mkern5\thinmuskip`. §456 C

708 Clean up the treatment of optional spaces in numerical specifications. §455 C

- *A construction like `2.5\space\space\dimen0` was previously valid after '`plus`' or '`minus`' only!*
- *I'm obviously working on Chapter 24 today.*

545 ↦ **709** Allow '`\font`' as a ⟨font identifier⟩ for the current font. §577 C

623 ↦ **710** Don't make `\gdef` global when *global_defs* < 0. §1218 C

711 Produce *zero_glue* as the outcome of `\advance\spaceskip by-\spaceskip`. §1229 E

712 Make `\show` do something appropriate for every possible token. §1294 I

559 ↦ **713** Replace the (single) `\tokens` parameter by an array of 256 token registers. §230 G

714 Allow `\indent` in math mode; also make `\valign` in math mode produce the '`Missing $`' error. §1046, 1093 C

715 Remove redundant code: There's no need to check *cur_group* or call *off_save* when starting alignments or equation numbers in displays. §1130, 1142 E

8 Jun 1983

716 Disallow `\openout-1` and `\closeout-1`. §1350 C

717 Disallow `\lastbox` in math mode. §1080 C

9 Jun 1983

718 Call *back_error*, not *error*, when `\leaders` aren't followed by proper glue. §1078 I

719 Initialize for a possible paragraph, after `\noalign` in a `\valign`. §785 D

10 Jun 1983

720 Expand the optional space after an ASCII constant. §442 C

12 Jun 1983

721 Set *space_factor* ← 1000 after a rule or a constructed accent. §1056, 1123 C

14 Jun 1983

722 Correct a serious blunder: Set *disc_width* ← 0 before testing if *s* is null (caught by JS). §870 D

- *This is a real bug that existed since the beginning! It showed up on page 37 of the Version 0* TRIP *manual, but I didn't notice the problem.*

708 ↦ **723** Make optional spaces after ⟨dimen⟩ like those after ⟨number⟩. §448 C

568 ↦ **724** Insert *every_display* before calling *build_page*. §1145 C

648 ↦ **725** Report TEX's capacity on overflow errors in a way that's fully consistent with other statistical reports. §42 I

17 Jun 1983

726 Make all `\tracing` decisions on the basis of ≥ versus <, not ≠ versus =. §581 C

- *Today I finished the draft of Chapter 27 (the last chapter)!*
- *The changes above were released as Version 0.99 on June 19, 1983.*

20 Jun 1983

727 Set `\catcode`\%=14` in `INITEX`. §232 C

587 ↦ **728** Call *normal_paragraph* when `\par` occurs in vertical mode. §1094 C

- *Once again I'm retiring about 8am and awaking about 4pm.*

21 Jun 1983

558 ↦ **729** Don't append an overfull rule solely because of `\hbadness`. §666 C

730 Don't allow the glue-ratio of shrinking to be less than −1. §810, 811 R

22 Jun 1983

653 ↦ **731** Declare the parameter to *print_roman_int* to be of type *integer*, instead of *nonnegative_integer* (found by Debby Clark). §69 B

690 ↦ **732** Make the keyword '`by`' optional (suggested by LL). §1236 C

24 Jun 1983

733 Say '`preloaded`' when announcing *format_ident*. §1328 I

25 Jun 1983

734 Add extra boxes and glue to the output of alignment. [This thwarts possible attempts at trickery by which system-dependent glue set values computed by `\span` could have gotten into TEX's registers by things like `\valign` and `\vsplit`. It also has the advantage of perfect accuracy in alignment of vertical rules.] §809 R

735 Make leaders affect the height or width of the enclosing boxes. §656, 671 C

- *Today I'm mainly installing a much-improved format for change files in* WEB *programs (suggested by DRF).*

28 Jun 1983

736 Permit `\unskip` in vertical mode when we know that it does nothing. §1106 C

1 Jul 1983

700 ↦ **737** Avoid redundant boxes when things like '`{\bf A}`' occur in math. §1186 E

738 Add a '`scaled`' feature to `\font` input. §1258 G

700 ↦ **739** Remember to correct *delta* when an accented box changes. §742 D

2 Jul 1983

740 Introduce *bypass_eoln*, to remove anomalous behavior on input files of length 1. (Suggested by DRF after the problem was discovered by LL). §31 R

4 Jul 1983

741 Allow codes like `^^b` as well as `^^B`. §352, 355 G

742 Introduce new parameters `\escapechar`, `\endlinechar`, `\defaulthyphenchar`, and `\defaultskewchar`, to make TEX less dependent on the character set. (This affects many modules, since a lot of error messages must be broken up so that they use *print_esc*.) G

7 Jul 1983

743 Use a system-dependent function *erstat* when opening or closing files (suggested by DRF). §27 P

11 Jul 1983

- *The computer is back up after more than 50 hours down time (due to air conditioning failure).*

744 Show total glue in the output of `\tracingpages`. §985 I

745 Guard against insertion into an hbox. §993 R

746 Legalize the assignment ⟨tokenvar⟩=⟨tokenvar⟩. §1227 C

747 Introduce a new parameter `\errhelp`. §1283 I

623 ↦ **748** Don't forget to check *global_defs* when `\tabskip` is changed. §782 F

12 Jul 1983

749 Allow an `\outer` macro to appear after `\string`, `\noexpand`, and `\meaning` (Todd Allen). §369, 471 C

750 Make '`\the`' an expandable control sequence (i.e., move it from the stomach to the throat); this cleans up several annoying glitches. §367 C

751 Allow `\unhbox` and `\unhcopy` in math mode if the box is void. §1110 C

13 Jul 1983

- *I lectured for four hours at the TUG meeting today after very little sleep!*

16 Jul 1983

- *The following were suggested by TUG meeting discussions.*

752 Round the value of *default_rule* more properly: It should be 26215. §463 L

700 ↦ **753** Fix `\mathaccent` again; it's still not right! The final height should be the maximum of the height of accented letter without superscript and the height of unaccented letter with superscript. §742 Q

754 Add a new feature `\newlinechar`. §59 G

755 Allow boxes and rules in discretionaries (suggested by somebody from Hewlett-Packard). §1121 G

756 Show all token expansions, not just macros, when `\tracingcommands`. §367 I

757 Allow `\char` in a `\hyphenation` list. §935 C

758 Introduce a new feature `\aftergroup`; it can be implemented with *save_stack*. §326 G

759 Run the running dimensions to alignment boundaries (suggested by ARK). §806 C

17 Jul 1983

760 Zero out *hyf* values at the edges, so that weird pattern data cannot lead to Pascal range checks. §965 R

761 Decrease the `hc` codes for hyphenation, so that code 127 cannot possibly be matched. §937, 962 R

672 ↦ **762** Allow whatsits after hyphenatable words. §899 C

604 ↦ **763** Represent an italic correction as an explicit kern. §1113 C

18 Jul 1983

764 Allow lowercase letters in file names. §519 C

765 Change the message '`No output file`' to: '`No pages of output`'. §642 I

766 Confirm that a quiet mode is being entered, when error interaction ends with `Q`, `R`, or `S` (suggested by ARK). §86 I

- *Version 0.999 was finally installed today; a new program listing has been printed.*

- *From now on, I plan to keep all section numbers unchanged.*
- *I'm done writing Appendix H; beginning to revise Chapter 20.*

25 Jul 1983

663 ↦ **767** Allow space after '`to`' in the `\read` command (FY). §1215 C

- *To bed at 1pm today.*

27 Jul 1983

665 ↦ **768** Stack the current type of `\if`; this precaution is necessary in general (FY). §498 S

- *To bed at 2pm today.*

29 Jul 1983

769 Avoid putting a control sequence in the hash table when it occurs after `\ifx`. (Requested by Math Reviews people.) §507 E

- *Finished a version of The TEXbook lacking only Appendices D, E, and I, for distribution to interested readers.*
- *To bed at 10:30pm, planning to arise regularly at 6am for a change.*

31 Jul 1983

766 ↦ **770** Call *update_terminal* when going quiet (HWT). §86 I

1 Aug 1983

771 Don't put an empty line at the end of an `\input` file! (This simplifies the rules and the program, and also gets around a bug that occurred at the end of files with *end_line_char* < 0.) §362 C

- *The changes above went into Version 0.9999, which was widely distributed.*

16 Aug 1983

665 ↦ **772** Rectify a ridiculous gaffe: I initialized q every time the loop of *change_if_limit* was performed! (Found by FY.) §497 B

648 ↦ **773** Distinguish '`string`' from '`strings`' when reporting statistics. §1334 I

774 Introduce lx, to correct a bug in `\xleader` computations (found by FY). §627 A

20 Aug 1983

775 Don't forget to apply \/ to ligatures! §1113 F

- *Today I began to read all previous issues of TUGboat, in preparation for Appendix D.*

27 Aug 1983

776 Add debugging hack number 16, to help catch subtle data structure bugs. §1339 I

777 Remove redundant setting and resetting of *name_in_progress*. §531 E

778 Suppress `\input` during a font size spec; otherwise *cur_name* is clobbered (found by MDS). §1258 S

779 Introduce new conditionals `\ifhbox` and `\ifvbox`. §505 G

29 Aug 1983

750 ↦ **780** Test for an empty list, if emptiness will mess up the data structure. (Found by Todd Allen.) §478 D

781 Use *fast_for_new_token* for efficiency. §466 E

782 Say '`has only`' instead of '`has`'. §579 I

- *These changes yield Version 0.99999, used only at Stanford.*

30 Aug 1983

783 Make funny blank spaces showable. §298 C

31 Aug 1983

754 ↦ **784** Make `\newlinechar` affect *print_char*, not just *print*. §58 C

4 Sep 1983

785 Add new features `\lastkern`, `\lastpenalty`, `\unkern`, `\unpenalty`. §424, 996, 1105 G

- *OK, Appendix D is finished!!*
- *The above changes have been installed in Version 0.999999.*

17 Sep 1983

548 ↦ **786** Don't bother making duplicate font identifiers; that was overkill, not really needed. §1258 P

- *Will this be the historic last change to TEX?*

18 Sep 1983

787 Correct a minor inconsistency, '`display`' not '`displayed`'. §211 I

20 Sep 1983

604 ↦ **788** Treat the kerns inserted for accents as explicit kerns. §1125 C

26 Sep 1983

789 Change '`log`' to '`transcript`' in several messages. §535, 1335 I

- *The index was finished today; I mailed the entire TEXbook East for final proofreading before publication.*

1 Oct 1983

790 Prevent uninitialized trie positions in case of overflow (found by Bernd Schulze). §944 D

7 Oct 1983

- *Henceforth our weekly 'TEX lunch' meetings will be called 'METAFONT lunch'.*
- *DRF begins to produce The TEXbook on our APS phototypesetter.*

14 Oct 1983

633 ↦ **791** Ignore spaces at the ends of lines also in `TEX.POOL` (found by DRF). §52 P

792 Initialize the *history* variable at *start_here* (DRF). §1332 D

18 Oct 1983

793 Extend *runaway* to catch runaway text (suggested by FY). §306 I

794 Reset *cur_cs* after *back_input*, not after scanning the '=' (found by FY). §1226 D

24 Oct 1983

638 ↦ **795** Change the error recovery for bad delimiters, in accordance with the changed syntax. (Found by Barry Smith.) §1183 I

9 Nov 1983

796 Optimize the code a bit more, based on empirical frequency data gathered during September and October: In §45, use the fact that the result is almost always true. In §380, delete '**while** *true* **do**' since many compilers implement that badly. Rewrite §852 to avoid calling *badness* in the most common case. §45, 380, 852 E

3 Dec 1983

797 Don't forget to call `error` after the message has been given (noticed by Gabi Kuper). §500 F

- *Version 1.0 released today incorporates all of the above.*

9 Dec 1983

- *Dinner party with 36 guests to celebrate TEX's coming of age.*

2 Feb 1984

786 ↦ **798** Reinstall `\font` precautions that I thought were unnecessary. I overlooked many problematic possibilities, like '`{\font\a=x \global\a} \the\font`' and '`\font\a=x \font\b=x \let\b=\undefined \the\a`', etc. (Found by Mike Urban.) The new remedy involves removal of the *font_ident* array and putting the identifiers into a frozen part of the hash table; so there's a sprinkling of

corrections in lots of modules. But basically the change is quite conservative, so it shouldn't spawn any new bugs (it says here). §222, 267, 1257 S

9 Feb 1984

799 Remove the possibility of double interrupt, in a scenario found by Clint Cuzzo. §1031 S

12 Feb 1984

800 Improve spacing in a formula like `$(A,<)$`. §764 Q

13 Feb 1984

801 Avoid a bad **goto**, as diagnosed by Clint Cuzzo and George O'Connor. (Must not go directly to *switch*.) §346 A

802 Conserve string pool space by not storing file name in two guises (suggested by DRF). §537 E

26 Feb 1984

803 Make scaled output look cleaner by printing fewer decimals whenever this involves no loss of accuracy. (Suggested by METAFONT development.) §103 I

2 Mar 1984

804 Maintain 17-digit accuracy, not 16; now constants like '`.00000762939453126pt`' will round correctly. §452 R

16 Mar 1984

805 Plug a loophole that permitted recursion in *get_next*, by disallowing deletions in *check_outer_validity*. §336 R

24 Mar 1984

806 Open the terminal before trying to wake it up, when the program starts bad. §1332 I

27 Mar 1984

807 Check that $k < 63$, to avoid the `\patterns{xxx...xxxdxxxdxxx}` anomaly found by Jacques Désarménien. §962 R

11 Apr 1984

808 Supply code for the missing case *adjust_node* in *copy_node_list*. §206 F

- *Yoicks, how could serious bugs like that have escaped detection?*

11 Jun 1984

627 ↦ **809** Initialize *char_base*, etc., for *null_font*. (Found by Nick Briggs.) §552 D

810 Clear the *buffer* array initially (Briggs). §331 R

21 Jun 1984

811 Look ahead for ligature or kern after a `\chardef`'d item (Désarménien). §1036 C

4 Jul 1984

812 Make the quarterword constraint explicit with a new '*bad*' case (19). §111 R

7 Jul 1984

813 Optimize *firm_up_the_line* slightly, to be consistent with the METAFONT program. §363 E

8 Jul 1984

814 Give additional diagnostics when `\tracingmacros>1`. §323 I

- *The changes above were incorporated in Version 1.1, released July 9, 1984.*

27 Jul 1984

815 Say '`see the transcript file`' after handling offline `\show` commands. (Suggested by METAFONT.) §1298 I

20 Oct 1984

816 Allow '0' in response to error prompts. §84 I

- *Those two changes led to Version 1.2.*

25 Nov 1984

817 Don't forget to check for *null* before looking at subfields of a node. (This was "dirty Pascal," with two quarterword 0's read as a halfword.) §846 R

818 Ditto in another place! §939 R

819 Remove the fixed-at-compile-time partition between lower and upper memory. §116, 125, 162 E

- *This major change in memory management completes Version 1.3, which was published in preliminary looseleaf form as 'TEX: The Program'.*

20 Dec 1984

820 Keep the *node_size* field from overflowing if the lower part of memory is too large. §125 R

- *That was another bug in existence from the beginning!*

5 Jan 1985

821 Improve the missing-format-file error (DRF). §524 I

7 Jan 1985

822 Update the terminal right away so that the welcoming message will appear as soon as possible (DRF). §61 I

23 Jan 1985

823 Convey more uncertainty in the help message at times of *confusion*. §95 I

824 Improve the *history* logic in the *warning_issued* case. §245 I

18 Feb 1985

810 ↦ **825** Stick to standard Pascal: Don't use *first* in a **for** loop. [Some procedures "threaten" it globally, according to British Standard 6192, section 6.8.3.9.] (Pointed out by CET.) §331 P

11 Apr 1985

826 Prevent nonexistent characters from being output by unusual combinations of ligatures and hyphenation. §915 S

15 Apr 1985

819 ↦ **827** Compute memory usage correctly in `INITEX`; the previous number was wrong because of a `WEB` text macro without parentheses (DRF). §164 L

16 Apr 1985

828 Speed up *flush_list* by not calling *free_avail* (DRF). §123 E

17 Apr 1985

788 ↦ **829** Introduce a special kind of kern for accent positioning; it must not disappear after a line break. §837, 879, 1125 A

18 Apr 1985

755 ↦ **830** Prevent `\lastbox` and `\unkern` from removing discretionary replacements. §1081, 1105 R

- *That completes Version 1.4.*

26 Apr 1985

831 Don't try *TEX_area* if a nonstandard file area has been specified (DRF). §537 C

- *That was #401 in TEX78; I never learn!*

30 Apr 1985

754 ↦ **832** Eliminate the limitation on `\write` length; the reason for it has disappeared (Nancy Tuma). §1370 C

8 May 1985

819 ↦ **833** Allocate two words for the head of the *active* list (CET). §162 D

11 May 1985

834 Change *wterm* to *wterm_ln* after a bad beginning (Bill Gropp). §1332 I

806 ↦ **835** Don't open the terminal twice (CET). §1332 E

22 May 1985

836 Test for *batch_mode* after trying to open the transcript file, not before (DRF). §92 R

837 Be prepared for string pool overflow while reading the command line! (This bug was first found in METAFONT, when it could occur more easily.) §525 R

7 Aug 1985

838 Fix a bug in `\edef\foo{\iffalse\fi\the\toks0}`: TEX should stay in the loop when expanding non-`\the`. (Found by Dan Brotsky.) §478 A

- *The above changes were incorporated in Version 1.5.*

27 Nov 1985

764 ↦ **839** Make '`plain`' a lowercase name, for consistency with the manual. §521 C

669 ↦ **840** Wake up the terminal for `\show` commands. §1294, 1297 I

- *The above changes were incorporated in Version 2.0, which was published as Volume B of the Computers & Typesetting series.*

15 Dec 1986

841 Punctuate the Poirot help message more carefully. §1283 I

28 Jan 1987

842 Make sure that *max_in_open* doesn't exceed 127 (DRF). §14 R

680 ↦ **843** Don't allow a `\kern` to be clobbered at the end of a pre-break list when a discretionary break is taken. (A missing '`else`' was the source of the error, diagnosed incorrectly before.) §881 D

844 Take account of discarded nodes when computing the background width after a discretionary. §840 D

- *That was the first really serious bug detected for more than 17 months! I found it while experimenting with right-to-left extensions.*
- *Version 2.1 was released on January 26, 1987.*

5 Feb 1987

845 Remove cases in *shorthand_def* that cannot occur (found by Pat Monardo). §1224 E

14 Apr 1987

846 Improve robustness of data structure display when debugging (Ronaldo Amá). §174, 182 R

21 Apr 1987

847 Make the storage allocation algorithm more elegant and efficient. §127 E

22 Apr 1987

742 ↦ **848** Calculate the empty-line condition properly when *end_line_char* is absent. §360 A

- *The previous three changes were found while I was teaching a class based on Volume B; they led to Version 2.2.*

28 Apr 1987

849 Avoid closing a file when TEX knows that it isn't open (JS). §560 E

3 Aug 1987

850 Clean up unfinished output if it's necessary to *jump_out* (Klaus Gunterman). §642 S

- *That makes Version 2.3; subsequent version numbers won't be logged here.*

19 Aug 1987

851 Indent rules properly in cases like `\hangindent=1pt$$\halign{...\cr\noalign{\hrule}}$$`. §806 A

20 Aug 1987

852 Introduce *co_backup* because of cases like `\hskip 0pt plus 1fil\ifdim` (Alan Guth). §366 S

9 Nov 1987

853 Change the calculation for number of leader boxes, so that it won't be too sensitive to roundoff error near exact multiples (M. F. Bridgland). §626 S

17 Nov 1987

854 Replace my stupid algorithm for fixed-point multiplication of negatives (W. G. Sullivan). §572 A

12 Dec 1987

855 Fix a typo in the initialization of hyphenation tables (Peter Breitenlohner). §952 B

- *That error was almost completely harmless, thus undetectable, except if some* `\lccode` *is 1 and no* `\patterns` *are given.*

23 Dec 1987

564 ↦ **856** Be more cautious when "relaxing" a previously undefined `\csname`; you might be inside a group (CET). §372 S

20 Apr 1988

857 Make sure *temp_head* is well-formed whenever it can be printed in a "runaway" message: Consider constructions like `\outer\def\a0{}\a\a` (Silvio Levy). §391 S

24 Apr 1988

858 Avoid conflicting use of the string pool in constructions like `\def\\#1{}\input a\\b` (Robert Messer). §260 S

10 May 1988

859 Amend the `\patterns` data structure when *trie_min* = 0 (Breitenlohner). §951, 953 R

25 May 1988

860 Guarantee that *trie_pointer* cannot be out of range. §923 R

861 Avoid additional bugs like #858 in constructions like `\input a\romannumeral1`, etc. §464, 465, 470 S

862 Prevent similar string pool confusion that could occur during the processing of `**\input\romannumeral6`. §525 R

19 Jun 1988

819 ↦ **863** Prevent a negative dividend from rounding upward, causing a loop (CET). §126 S

819 ↦ **864** Adopt a smoother allocation strategy when memory is nearly gone (CET). §126 E

20 Jun 1988

852 ↦ **865** Initialize *cur_order*, now that it's being backed up (Tsunetoshi Hayashi). §439 D

6 Nov 1988

612 ↦ **866** Disable *fatal_error* in *prompt_input*, so that *open_log_file* can use it safely (Tim Morgan). §71 S

836 ↦ **867** Force terminal output whenever *open_log_file* fails. §535 S

- *We're now up to Version 2.94; I sincerely hope all bugs have been found.*

REFERENCES

1. Piet Hein, *Grooks*, MIT Press, 1966.
2. C. Széchy, *Foundation Failures*, Concrete Publications, London, 1961.
3. A. Endres, 'An analysis of errors and their causes in system programs', *Proc. Int. Conf. Software Eng.*, 1975, pp. 327–336.
4. Victor R. Basili and Barry T. Perricone, 'Software errors and complexity: an empirical investigation', *Communications of the ACM*, **27**, 42–52 (1984).
5. L. A. Belady and M. M. Lehman, 'A model of large program development', *IBM Systems J.*, **15**, 225–252 (1976).
6. Donald E. Knuth, *TEX: The Program*, Addison-Wesley, 1986.
7. Donald E. Knuth, 'Literate programming', *The Computer Journal*, **27**, 97–111 (1984).
8. Donald E. Knuth, 'The WEB system of structured documentation', *Stanford Computer Science Report STAN-CS-980*, September 1983.
9. Patrick Winston, *Artificial Intelligence: An MIT Perspective*, MIT Press, 1979.
10. Donald E. Knuth, 'The letter S', *The Mathematical Intelligencer*, **2**, 114–122 (1980).
11. Donald E. Knuth, 'Mathematical typography', *Bulletin of the American Mathematical Society* (new series) **1**, 337–372 (1979).
12. Donald E. Knuth, *Seminumerical Algorithms*, second edition, Addison-Wesley, 1981.
13. Donald E. Knuth and Michael F. Plass, 'Breaking paragraphs into lines', *Software—Practice and Experience*, **11**, 1119–1184 (1981).
14. Donald E. Knuth, 'The concept of a meta-font', *Visible Language*, **16**, 3–27 (1982).
15. Donald E. Knuth, *The TEXbook*, Addison-Wesley, 1984.
16. Barbara Beeton (ed), *TEX and METAFONT: Errata and Changes, 09 September 1983*, distributed with *TUGboat*, **4** (1983).
17. Donald E. Knuth, *TEX, a System for Technical Text*, American Mathematical Society, 1979.
18. Donald E. Knuth, *TEX and METAFONT: New Directions in Typesetting*, Digital Press, 1979.
19. David R. Fuchs and Donald E. Knuth, 'Optimal prepaging and font caching', *ACM Transactions on Programming Languages and Systems*, **7**, 62–79 (1985).
20. Donald E. Knuth, 'A torture test for TEX', *Stanford Computer Science Report STAN-CS-1027*, November 1984.
21. Donald E. Knuth, 'A torture test for METAFONT', *Stanford Computer Science Report STAN-CS-1095*, January 1986.
22. Donald E. Knuth, *Sorting and Searching*, Addison-Wesley, 1973.
23. Brian W. Kernighan and Lorinda L. Cherry, 'A system for typesetting mathematics', *Communications of the ACM*, **18**, 151–157 (1975).
24. Guy L. Steele Jr., Donald R. Woods, Raphael A. Finkel, Mark R. Crispin, Richard M. Stallman and Geoffrey S. Goodfellow, *Hacker's Dictionary: A Guide to the World of Wizards*, Harper and Row, 1983.
25. Donald E. Knuth, *Fundamental Algorithms*, Addison-Wesley, 1968.
26. Reinhard Budde, Christiane Floyd, Reinhard Keil-Slawik and Heinz Züllighoven, (eds) *Software Development and Reality Construction*, in preparation.