CC/NUMBER 40
OCTOBER 3, 1988

This Week's Citation Classic®

Dijkstra E W. *A discipline of programming.* Englewood Cliffs, NJ: Prentice-Hall, 1976. 217 p.
[Technological University, Eindhoven, The Netherlands]

This was the first book to treat the programming task as a mathematical challenge. It shows how to derive a program from its functional specification by first choosing the structure of the proof that will demonstrate the correctness of the program under design. [The *SCI*® and *SSCI*® indicate that this book has been cited in over 260 publications.]

Edsger W. Dijkstra
Department of Computer Sciences
University of Texas
Austin, TX 78712-1188

May 8, 1988

In the early 1970s I knew I had to forge programming into an effective mathematical discipline and got my first glimpses of how to do that. This vision clashed with that of my Department of Mathematics at the Eindhoven University of Technology, which I left in 1973 to become a Burroughs Research Fellow (with the generous charter "to do my own thing"). Then things moved quickly.

At the Blanchland meeting of the International Federation for Information Processing Working Group 2.3 on "Programming methodology," during the night of October 23-24, 1973, all the pieces fell into place. That evening I could not sleep and found myself preparing my lecture; with my brain burning, I left my bed at 2:30 a.m. and wrote for more than an hour.

It was not only my former department that was not ready for it. When, eight months later, I submitted a paper under the title "Guarded commands, nondeterminacy and a calculus for the derivation of programs" to the *Communications of the ACM*, one referee objected so strongly that I had to remove "a calculus" from the title. This paper[1] presented the bare ingredients; the monograph *A Discipline of Programming* developed the methodology and applied it to many programming problems.

The book embodied four novelties, two technical ones and two stylistic ones. One technical novelty was to define programming language semantics by means of a predicate transformer for the weakest precondition. Its other technical novelty was the inclusion of nondeterminacy, thereby smoothing the way to treat uni- and multiprogramming on the same footing. One stylistic novelty was the introduction of "guarded commands," a highly elegant notational device that is equally applicable to alternative and repetitive programming constructs and thus became the preferred vehicle for many researchers in the field. Its other stylistic novelty was its stress on methodology: instead of offering problems and solutions, it offered problems and then discussed how to solve them.

Now, more than a decade after its publication, it is still a standard work that sells accordingly. It is also dated. In the meantime we have gained a lot of experience in carrying out the type of formal manipulations required, and, thanks to a few (minor but vital) notational innovations, we have become much more effective in the application of the predicate calculus, which is now the modern programmer's indispensable tool for his daily reasoning.[2-3]

1. **Dijkstra E W.** Guarded commands, nondeterminacy and formal derivation of programs. *Commun. ACM* 18:453-7, 1975. (Cited 85 times.)
2. **Gries D.** *The science of programming.* New York: Springer-Verlag, 1981. 366 p. (Cited 40 times.)
3. **Reynolds J C.** *The craft of programming.* Englewood Cliffs, NJ: Prentice/Hall International, 1981. 434 p. (Cited 10 times.)

©1988 by ISI® CURRENT CONTENTS®

IA - 16