FOREWORD

Edsger W. Dijkstra is one of a very small number of people who, through their research and teaching, have provided computing with an intellectual foundation that can justifiably be termed a science. His seminal contributions to such fields as language and compiler design, operating systems, and programming methodology, have led to his receiving numerous honours and awards, including the 1972 ACM Turing Award, recognized by all as the highest award in computing.

This book is another such expression of the debt that our subject owes to Edsger. It has been prepared, as a surprise present for him on the occasion of his sixtieth birthday, by some of the many people whose own contributions to our field have been inspired or assisted by him. In view of the galaxy of talent represented here by this book's collective authorship, and the great personal debt that I owe to Edsger, I feel doubly privileged to have been invited to write a foreword.

I have had the pleasure of knowing Edsger for almost thirty years. I first met him when he lectured in Brighton on the Algol 60 compiler that he and his colleague J.A. Zonneveld had just produced for the Electrologica X-1 computer. Their compiler, it should be noted, had been completed in August 1960, just seven months after they had received a copy of the Algol-60 Report. This was long before anyone else had learnt how to cope with all the problems and opportunities that the new language provided.

A year or so later, in 1963, I asked his opinion on the suggestion that my colleague Lawford Russell and I write a book about the Algol 60 compiler that we had produced. This compiler, for the English Electric KDF9 computer, was based in no small measure on the design of his original X-1 compiler. Edsger was very supportive, but gave us some very valuable, albeit uncomfortable, advice: instead of just describing our compiler, we should try to describe all the alternatives we had considered for each design choice and the basis on which we had chosen amongst these alternatives. Moreover, he advised that we should make a point of admitting it when our choice had been arbitrary and/or had in retrospect proved to be wrong. I have always been extremely grateful for this advice - and have taken care to pass it on to all my graduate students.

For a number of years after, IFIP Working Group 2.1 on Algol provided a regular opportunity for me to enjoy and profit from his company (until we and a number of other members resigned over Algol 68). Thereafter it was mainly meetings of IFIP Working Group 2.3 on Programming Methodology, and Newcastle's Annual International Seminar on the Teaching of Computing Science, that provided me further such opportunities.

One final personal note about my interactions with Edsger concerns the 1968 NATO Software Engineering Conference. We have both since gone on record as to how the discussions at this conference on the "software crisis", and on the potential for software-induced catastrophes, strongly influenced our thinking and our subsequent research activities. In Edsger's case it led him into an immensely fruitful long-term study of the problems of producing high quality programs. In my own case, it led me to consider the then very novel, and still somewhat controversial, idea of design fault tolerance. Suffice it to say that our respective choices of research problem suitably reflect our respective skills at program design and verification.

Edsger's 1963 advice to me is of course in line with the fact that he has always himself set and followed very high standards for clarity and presentation of writing and lecturing. Although the lecturing style that he has developed has its

critics (some of whom mistakenly interpret his reflective pauses as mere theatricality), my own regret is that I cannot match either his clarity or his skill in inventing or choosing suitable examples.

The problems he chooses to concentrate on are always very skillfully selected (e.g. the dining philosophers problem, on-the-fly garbage collection), and many have become famous standards by which subsequent research is judged. The vivid and effective visual metaphors he still uses serve as a reminder of the effectiveness of the excellent actual pictures he invented many years ago to describe, for example, the process of translating arithmetic expressions to post-fix form, and the problem of deadlocks amongst processes competing for shared resources.

Edsger is, with all that the word implies, a perfectionist, who expects as much of his listeners and readers as he demands of himself. His programming and his mathematics are strongly guided by his concern for clarity of notation and exposition, and indeed for what he quite justifiably terms 'beauty'. Thus his descriptions of problems and solutions, both in his lectures and published papers, and in his EWD series of documents[*], distributed via a network of friends and colleagues in a fashion reminiscent of Russian "samizdat" literature, are often vivid and compelling.

Although over the years much of his work has had a very immediate impact, on debate if not always in practice, some of his more recent diagnoses and prescriptions have proved harder to take. This is in part because he is sometimes more concerned with the truth of his arguments than with whether they are couched in terms that will help to ensure that they have the desired effect on his audience. Nevertheless, careful study of all his writings is highly recommended to all who care for the future health of computing science.

His most recent work, as represented by his continuing series of EWD documents, is mainly on the effective structure of logical arguments, applied to mathematics as much as programming - a distinction whose validity he would deny, since to him programming *is* mathematics. This work sets standards that many cannot even recognise, let alone aspire to. However I am confident that it will eventually have deep and long-lasting effects, much of it indirect, through the inspiration that it is providing to close colleagues.

The present book is, in effect, part of this dissemination process, since it contains many examples of the inspiration he has provided to his numerous "disciples". Thus they, and I, hope that this book will not just serve as an affirmation of the debt all computing scientists already owe to Edsger's work and teaching. Rather, we hope that it will also help to explain to readers who do not have close personal contact with Edsger the continuing impact of his current work.

Let me end with a quotation from George Bernard Shaw:

> The reasonable man adapts himself to the world: the unreasonable one persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable man.

Edsger W. Dijkstra is, in many ways, just such a man. Needless to say, the world of computing science could well do with many more "unreasonable" men (and women) of his calibre!

---

[*] A superb collection of these EWD documents, which includes several characteristically acerbic and entertaining trip reports on his experiences at Newcastle Seminars, is to be found in "Selected Writings on Computing: A Personal Perspective", by E.W. Dijkstra. Springer Verlag, New York, 1982.

Brian Randell

Newcastle upon Tyne

25 September 1989