An excerpt from *An Approach to Computer Language Design*, W. M. McKeeman, PhD thesis, 1966. Apologies to the ladies for the consistent use of masculine pronouns.

The Goals of Computer Language Design

The universe and its reflection in the idea of man have wonderfully complex structures. Our ability to comprehend this complexity and perceive an underlying simplicity is intimately bound with our ability to symbolize and communicate our experience. The scientist has been free to extend and invent language whenever old forms became unwieldy or inadequate to express his ideas. His readers however have faced the double task of learning his new language and the new structures he described. There has therefore arisen a natural control: work of elaborate linguistic inventiveness and meager results will not be widely read.

As the computer scientist represents and manipulates information within a machine, he is simulating to some extent his own mental processes. He must, if he is to make substantial progress, have linguistic constructs capable of communicating arbitrarily complicated information structures and processes to his machine. One might expect the balance between linguistic elaboration and achieved results to be operable. Unfortunately, the computer scientist, before he can obtain his results, must successfully teach his language to one particularly recalcitrant reader: the computer itself. This teaching task, called compiler writing, has been formidable.

Consequently, the computing community has assembled, under the banner of standardization, a considerable movement for the acceptance of a few committee-defined languages for the statement of all computer processes. The twin ideals of a common language for programmers and the immediate interchangibility of programs among machines have largely failed to materialize. The main reason for the failure is that programmers, like all scientists before them, have never been wholly satisfied with their heritage of linguistic constructs. We hold that the demand for a fixed standard programming language is the antithesis of a desire for progress in Computer Science. That the major responsibility for computer language design should rest with the language user will be our central theme.